MetaSensing: Intelligent Metasurface Assisted RF 3D Sensing by Deep Reinforcement Learning

Jingzhi Hu, *Graduate Student Member, IEEE*, Hongliang Zhang, *Member, IEEE*,
Kaigui Bian, *Senior Member, IEEE*, Marco Di Renzo, *Fellow, IEEE*, Zhu Han, *Fellow, IEEE*,
and Lingyang Song, *Fellow, IEEE*

***Abstract*— Using RF signals for wireless sensing has gained increasing attention. However, due to the unwanted multi-path fading in uncontrollable radio environments, the accuracy of RF sensing is limited. Instead of passively adapting to the environment, in this paper, we consider the scenario where an intelligent metasurface is deployed for sensing the existence and locations of 3D objects. By programming its beamformer patterns, the metasurface can provide desirable propagation properties. However, achieving a high sensing accuracy is challenging, since it requires the joint optimization of the beamformer patterns and mapping of the received signals to the sensed outcome. To tackle this challenge, we formulate an optimization problem for minimizing the cross-entropy loss of the sensing outcome, and propose a deep reinforcement learning algorithm to jointly compute the optimal beamformer patterns and the mapping of the received signals. Simulation results verify the effectiveness of the proposed algorithm and show how the size of the metasurface and the target space influence the sensing accuracy.**

***Index Terms*— RF 3D sensing, metasurface, deep reinforcement learning, policy gradient algorithm, beamformer pattern design.**

I. INTRODUCTION

RECENTLY, leveraging widespread radio-frequency (RF) signals for wireless sensing applications has attracted growing research interest. Different from methods based on wearable devices or surveillance cameras, RF sensing techniques need no direct contact with the sensing targets [1]. The basic principle behind RF sensing is that the influence of the target objects on the propagation of wireless signals can be potentially recognized by the receivers [2]. RF sensing techniques can be widely applied to many scenarios of daily life, such as surveillance [3], crowd sensing [4], ambient assisted living [5], and remote health monitoring [6]. In these applications, it is crucial to have high sensing accuracies.

Many RF-based sensing methods based on WiFi signals or millimeter wave signals have been proposed for sensing and recognizing human being and objects. In [7], the authors designed an RF sensing system that can detect the location and type of moving objects by using WiFi signals. In [8], the authors proposed a deep learning based RF sensing framework that can remove environmental and subject-specific information and can extract environmental/subject-independent features contained in the sensing data. In [9], the authors designed a low-power RF sensing system that automatically collects the behavioral patterns of people.

In addition, using RF sensing to capture human beings and indoor scenes has being explored. In [10], [11], the authors used wide-band RF transceivers with multiple-input-multiple-output (MIMO) antennas to capture images of human skeletons and showed that it is possible to reconstruct the human skeleton even when the RF signals are blocked by walls. In [12], the authors proposed to use mutually orthogonally coded millimeter wave signals to image the scenes including human beings and objects. However, using RF signals for sensing usually encompasses a signal collection and analysis process that assumes the radio channel environment to be fixed. The radio environment is unpredictable and usually unfavorable, and thus the sensing accuracy of conventional RF sensing methods is usually affected by unwanted multi-path fading [13], [14], and/or unfavorable propagation channels from the RF transmitters to the receivers.

Intelligent metasurfaces have been proposed as a promising solution for turning unwanted propagation channels into favorable ones [15], [16]. A metasurface is composed of a large number of electrically reconfigurable elements, which applies different phase-shifts to the RF signals that impinge upon it [17], [18]. By programming the reconfigurable elements, a metasurface deployed in the environment can change the RF propagation channel and create favorable signal beams

---

Manuscript received July 15, 2020; revised October 14, 2020; accepted November 23, 2020. Date of publication May 10, 2021; date of current version June 17, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61931019, Grant 61829101, Grant 61625101, and Grant 61941101; in part by the NSF under Grant EARS-1839818, Grant CNS-1717454, Grant CNS-1731424, and Grant CNS-1702850; and in part by the European Commission through the H2020 ARIADNE project under grant agreement number 871464 and through the H2020 RISE-6G project under grant agreement number 101017011. *(Corresponding author: Lingyang Song.)*

Jingzhi Hu and Lingyang Song are with the Department of Electronics, Peking University, Beijing 100871, China (e-mail: jingzhi.hu@pku.edu.cn; lingyang.song@pku.edu.cn).

Hongliang Zhang is with the Department of Electrical Engineering, Princeton University, Princeton, NJ 08540 USA (e-mail: hongliang.zhang92@gmail.com).

Kaigui Bian is with the Department of Computer Science, Peking University, Beijing 100871, China (e-mail: bkg@pku.edu.cn).

Marco Di Renzo is with the Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des Signaux et Systèmes, 91192 Gif-sur-Yvette, France (e-mail: marco.di-renzo@universite-paris-saclay.fr).

Zhu Han is with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77004 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul 446-701, South Korea (e-mail: hanzhu22@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSAC.2021.3078492.

Digital Object Identifier 10.1109/JSAC.2021.3078492

for sensing [19]. We refer to the coding of the reconfigurable elements as the *beamformer patterns of the metasurface*. Through dynamically designing the beamformer patterns, a metasurface can actively control the RF signal beams in the sensing process, which potentially improves the sensing accuracy. Instead of employing complex and sophisticated RF transmitters and receivers [20], metasurface-assisted RF sensing paves a new way of developing RF sensing methods, which have the capabilities of controlling, programming, and hence customizing the wireless channel. The authors of [21] explored the use of metasurfaces to assist the RF sensing and to obtain 2D images for human beings. In [22], the authors proposed a metasurface-assisted RF system to obtain localization information of mobile users. Nevertheless, no research works have tackled the analysis and design of metasurface-assisted 3D RF sensing, which is more challenging to analyze and optimize than 2D RF sensing.

In this paper, we consider a metasurface-assisted RF 3D sensing scenario, which can sense the existence and locations of 3D objects in a target space. Specifically, by programming the beamformer patterns, the metasurface performs beamforming and provides desirable RF propagation properties for sensing. However, there are two major challenges in obtaining high sensing accuracy in metasurface-assisted RF sensing scenarios.

- *First*, the beamformer patterns of the metasurface need to be carefully designed to create favorable propagation channels for sensing.
- *Second*, the *mapping of the received signals*, i.e., the mapping from the signals received at the RF receiver to the sensing results of the existence and locations of the objects, needs to be optimized as well.

Nevertheless, the complexity of finding the optimal beamformer patterns is extremely high because the associated optimization problem is a discrete nonlinear programming with a large number of optimization variables. Besides, the optimization of the beamformer patterns and the mapping of the received signals are closely coupled together, which makes optimizing the sensing accuracy in metasurface-assisted RF sensing scenarios even harder.

To tackle these challenges, we formulate an optimization problem for sensing accuracy maximization by minimizing the cross-entropy loss of the sensing results with respect to the beamformer patterns and the mapping of the received signals. In order to solve the problem efficiently, we formulate a Markov decision process (MDP) for the optimization problem and propose a deep reinforcement learning algorithm. The proposed deep reinforcement learning algorithm is based on the policy gradient algorithm [23] and is referred to as the progressing reward policy gradient (PRPG) algorithm, since the reward function of the MDP is consistently improved during the learning process. The computational complexity and the convergence of the proposed algorithm are analyzed. Moreover, we derive a non-trivial lower-bound for the sensing accuracy for a given set of beamformer patterns of the metasurface. Simulation results verify the effectiveness of the proposed algorithm and showcase interesting performance trends about the sensing accuracy with respect to the size of the metasurface and the target space. In particular, the contributions of this paper can be summarized as follows.

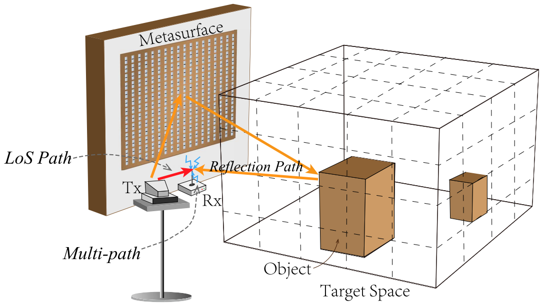

Fig. 1. Illustration of the metasurface-assisted RF sensing scenario.

- We consider a metasurface-assisted RF sensing scenario which can sense the existence and locations of objects in a 3D space. Then, we formulate an optimization problem to minimize the cross-entropy loss of the sensing results through optimizing the beamformer patterns and the mapping of the received signals. To this end, we adopt an MDP-based framework.
- We propose a deep reinforcement learning algorithm named PRPG to solve the formulated MDP. The complexity and the convergence of the proposed algorithm are analyzed, and a non-trivial lower-bound for the sensing accuracy is derived.
- We use simulation results to verify that the proposed algorithm outperforms other benchmark algorithms in terms of training speed and sensing accuracy. The simulation results unveil trends about the sensing accuracy as a function of the size of the metasurface and the target space, which gives insights on the implementation of practical metasurface-assisted RF sensing systems.

The rest of this paper is organized as follows. In Section II, we introduce the model of the metasurface-assisted RF sensing scenario. In Section III, we formulate the optimization problem to optimize the sensing accuracy by minimizing the cross-entropy loss of the sensing results. In Section IV, we formulate an MDP for the optimization problem and then propose the PRPG algorithm to solve it. In Section V, the complexity and convergence of the PRPG algorithm are analyzed, and a lower-bound for the sensing accuracy is derived. Simulation results are provided in Section VI and conclusions are drawn in Section VII.

II. SYSTEM MODEL

In this section, we introduce the metasurface-assisted 3D RF sensing scenario, which is illustrated in Fig. 1. In this scenario, there exists a pair of single-antenna RF transceivers, a metasurface, and a target space where the objects are located. The metasurface reflects and modifies the incident narrow-band signals at a certain frequency f_c. The Tx unit and Rx unit of the transceiver transmits and receives at f_c, respectively. The target space is a cubical region that is discretized into M equally-sized *space grids*. Each space grid is of size $\Delta l_x \times \Delta l_y \times \Delta l_z$.

The sensing process adopted in the considered scenario can be briefly described as follow. The signals transmitted by the

Tx unit are reflected and modified by the metasurface before entering into the target space. The modified signals are further reflected by the objects in the target space and received by the Rx unit. Then, the Rx unit maps the received signals to the sensing result, which indicates whether an object exists in each space grid.

In the following, we introduce the metasurface model in Subsection A, the channel model accounting for the metasurface in Subsection B, and the sensing protocol in Subsection C.

A. Metasurface Model

A metasurface is an artificial thin film of electromagnetic reconfigurable material, which is composed of uniformly distributed reconfigurable elements [24]. As shown in Fig. 1, the reconfigurable elements of the metasurface are arranged in a two-dimensional array. By controlling the positive-intrinsic-negative (PIN) diodes coupled with each reconfigurable element, the reconfigurable element can adjust its electromagnetic response to the incident RF signals. For each reconfigurable element, we refer to the different responses to incident RF signals as the reconfigurable element's *configuration* as in [25]. By changing the configuration of each reconfigurable element, the metasurface is able to modify the reflected signals and perform beamforming [26].

We assume that each reconfigurable element has N_S configurations, and each configuration of an element has a unique reflection coefficient for the incident RF signals. To be specific, we assume that each row and column of the metasurface contain the same number of reconfigurable elements, and the total number of reconfigurable elements is denoted by N. Based on [27], we denote the reflection coefficient of the n-th reconfigurable element corresponding to the incident signal from the TX unit and the reflected signal towards the m-th space grid by $r_{n,m}(c_n)$. Here, $c_n \in [1, N_S]$ denotes the configuration of the n-th reconfigurable element and $c_n \in \mathbb{Z}$, where $\mathbb{Z}$ denotes the set of integers.

B. Channel Model

In the considered metasurface-assisted RF sensing scenario, the Tx unit and Rx unit are equipped with a single antenna to transmit and receive RF signals. The Tx antenna is a directional antenna, which points towards the metasurface so that most of the transmitted signals are reflected by the metasurface and propagate into the target space. The signals reflected by the metasurface are reflected by the objects in the target space and then reach the Rx antenna. The Rx antenna is assumed to be omni-directional and located right below the metasurface, as shown in Fig. 1. This setting ensures that the signals reflected by the metasurface are not directly received by the Rx antenna, and thus most of the received signals contain the information of the objects in the target space.

As shown in Fig. 1, the transmission channel from the Tx antenna to the Rx antenna comprises three types of paths, i.e., the line-of-sight (LoS) path, the reflection paths, and the environmental scattering paths. The LoS path is referred to the direct signal path from the Tx antenna to the Rx antenna. The reflection paths are the paths from the Tx antenna to the Rx antenna via the reflections from the metasurface and the objects in the target space. The environmental scattering paths account for the signals paths between the Tx antenna and the Rx antenna which encompass complex reflection and scattering in the surrounding environment. Then, the equivalent baseband representation of the received signal containing the signals from all these three types of paths is denoted by y and can be expressed as

$$y = h_{\mathrm{los}} \cdot \sqrt{P} \cdot x + \sum_{m=1}^{M} \sum_{n=1}^{N} h_{n,m}(c_n, \nu_m) \cdot \sqrt{P} \cdot x + h_{\mathrm{rl}} \cdot \sqrt{P} \cdot x + \sigma, \quad (1)$$

where P is the transmit power, and x denotes the transmitted symbol.

The terms in (1) can be explained in detail as follows. The first term, i.e., $h_{\mathrm{los}} \cdot P \cdot x$, corresponds to the signal received in the LoS path, where h_{los} denotes the channel gain. Based on [28], h_{los} can be expressed as

$$h_{\mathrm{los}} = \frac{\lambda}{4\pi} \cdot \frac{\sqrt{g_T g_R} \cdot e^{-j2\pi d_{\mathrm{los}}/\lambda}}{d_{\mathrm{los}}}, \quad (2)$$

where λ is the wavelength of the signal, g_T and g_R denote the gains of the Tx and Rx antennas, respectively, and d_{los} is the distance from the Tx antenna to the Rx antenna.

The second term in (1) corresponds to the signals that reach the Rx antenna via $N \cdot M$ reflection paths. In the second term, $h_{n,m}(c_n, \nu_m)$ denotes the gain of the reflection path via the n-th reconfigurable element in configuration c_n and the m-th space grid with reflection coefficient ν_m. Based on [26], [29], $h_{n,m}(c_n, \nu_m)$ can be formulated as follows

$$h_{n,m}(c_n, \nu_m) = \frac{\lambda^2 \cdot r_{n,m}(c_n) \cdot \nu_m \cdot \sqrt{g_T g_R} \cdot e^{-j2\pi(d_n + d_{n,m})/\lambda}}{(4\pi)^2 \cdot d_n \cdot d_{n,m}}, \quad (3)$$

where d_n denotes the distance from the Tx antenna to the n-th reconfigurable element and $d_{n,m}$ denotes the distance from the n-th reconfigurable element to the Rx antenna via the center of the m-th space grid.

Finally, the third and fourth terms in (1) correspond to the signals from the environmental scattering paths and the additive noise at the Rx antenna, respectively. The symbol $h_{\mathrm{rl}} \in \mathbb{C}$ denotes the equivalent gain of all the environmental scattering paths, and σ is a random signal that follows the complex normal distribution, $\sigma \sim \mathcal{CN}(0, \epsilon)$ with ϵ being the power of the noise.

Moreover, we refer to the vector of configurations selected for the N reconfigurable elements as a *beamformer pattern* of the metasurface, which can be represented by an $N \times N_S$-dimensional binary row vector $\boldsymbol{c} = (\hat{\boldsymbol{o}}(c_1), \ldots, \hat{\boldsymbol{o}}(c_N))$. Specifically, $\hat{\boldsymbol{o}}(i)$ ($\forall i \in [1, N_S]$) denotes the N_S-dimensional row vector whose i-th element is 1 and the other elements are 0. Based on the definition of the beamformer pattern, the received signal in (1) can be reformulated as

$$y = h_{\mathrm{los}} \cdot \sqrt{P} \cdot x + \boldsymbol{c}\boldsymbol{A}\boldsymbol{\nu} \cdot \sqrt{P} \cdot x + h_{\mathrm{rl}} \cdot \sqrt{P} \cdot x + \sigma, \quad (4)$$

where $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_M)$ denotes the vector of reflection coefficients of the M space grids, $\boldsymbol{A} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_M)$ is referred

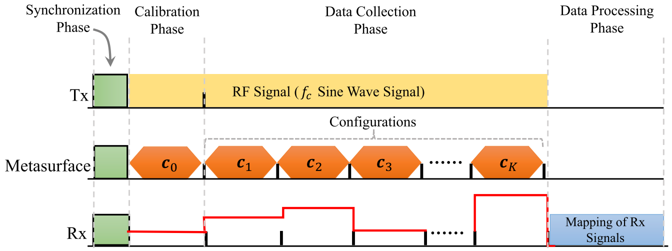

Fig. 2. A cycle of the RF sensing protocol.

to as the *projection matrix*, and $\boldsymbol{\alpha}_m = (\hat{\boldsymbol{\alpha}}_{m,1}, \ldots, \hat{\boldsymbol{\alpha}}_{m,N})^T$ with $\hat{\boldsymbol{\alpha}}_{m,n} = (\hat{\alpha}_{m,n,1}, \ldots, \hat{\alpha}_{m,n,N_S})$. Also, for all $m \in [1, M]$, $n \in [1, N]$, and $i \in [1, N_S]$, $\hat{\alpha}_{m,n,i}$ denotes the channel gain of the reflection path via the n-th reconfigurable element in configuration i and the m-the space grid with a unit reflection coefficient, which can be expressed as follows based on (3).

$$\hat{\alpha}_{m,n,i} = \frac{\lambda^2 \cdot r_{n,m}(i) \cdot \sqrt{g_T g_R}}{(4\pi)^2 d_n d_{n,m}} \cdot e^{-j2\pi(d_n + d_{n,m})/\lambda}. \tag{5}$$

C. RF Sensing Protocol

To describe the RF sensing process in the metasurface-assisted scenario clearly, we formulate the following *RF sensing protocol*. In the protocol, the timeline is slotted and divided into *cycles*, and the Tx unit, the Rx unit, and the metasurface operate in a synchronized and periodic manner. As shown in Fig. 2, each cycle consists of four phases: a *synchronization phase*, a *calibration phase*, a *data collection phase*, and a *data processing phase*. During the synchronization phase, the Tx unit transmits a synchronization signal to the metasurface and to the Rx unit, which identifies the start time of a cycle.

Then, in the calibration phase, the Tx unit transmits a narrow band constant signal, i.e., symbol x, at frequency f_c. The metasurface sets the beamformer pattern to be $\boldsymbol{c}_0 = (\hat{\boldsymbol{o}}(1), \ldots, \hat{\boldsymbol{o}}(1))$, i.e., the N reconfigurable elements are in their first/default configuration. Besides, the received signal of the Rx unit is y_0.

The data collection phase is divided into K *frames* that are evenly spaced in time. During this phase, the Tx unit continuously transmits the narrow band RF signal, while the metasurface changes its beamformer pattern at the end of each frame. As shown in Fig. 2, we denote the beamformer patterns of the metasurface corresponding to the K frames by the binary row vectors $\boldsymbol{c}_1, \ldots \boldsymbol{c}_K$. Specifically, the K beamformer patterns of the metasurface during the data collection phase constitutes the *control matrix*, which is denoted by $\boldsymbol{C} = (\boldsymbol{c}_1^T, \ldots, \boldsymbol{c}_K^T)^T$. Besides, as $\boldsymbol{c}_k$ is a binary row vector, control matrix is a binary matrix.

To remove the signal from the LoS path which contains no information of the target space, the received signals in the K frames are subtracted by y_0. The K differences constitute the *measurement vector*, which is a noisy linear transformation of $\boldsymbol{\nu}$ by the matrix $\boldsymbol{\Gamma}$, i.e.,

$$\tilde{\boldsymbol{y}} = \boldsymbol{y} - \boldsymbol{y}_0 = \boldsymbol{\Gamma}\boldsymbol{\nu} + \tilde{\boldsymbol{\sigma}}, \tag{6}$$

where $\boldsymbol{\Gamma} = \sqrt{P} \cdot x \cdot (\boldsymbol{C} - \boldsymbol{C}_0)\boldsymbol{A}$ with $\boldsymbol{C}_0 = (\boldsymbol{c}_0^T, \ldots, \boldsymbol{c}_0^T)^T$, $\boldsymbol{y}$ is a K-dimensional vector consisting of the sampled received signals during the K frames that can be calculated by (4), $\boldsymbol{y}_0$ is a K-dimensional vector with all the elements being y_0, and $\tilde{\boldsymbol{\sigma}}$ is the difference between the noise signals and the environmental scattering signals, i.e., $\boldsymbol{y}$ and $\boldsymbol{y}_0$. We assume that the environment in the considered scenario is static or changing slowly. In this case, the signals from the environmental scattering paths, i.e., $h_{\mathrm{rl}} \cdot \sqrt{P} \cdot x$ is subtracted in (6), and $\tilde{\boldsymbol{\sigma}}$ contains the difference between the Gaussian noise signals of $\boldsymbol{y}$ and $\boldsymbol{y}_0$.[1] Specifically, the k-th element of $\tilde{\boldsymbol{\sigma}}$ is $\tilde{\sigma}_k \sim \mathcal{CN}(0, 2\epsilon)$. We refer to $\tilde{\boldsymbol{y}}$ as the *measurement vector*. Since $\boldsymbol{\Gamma}$ determines how the reflection characteristics of the objects are mapped to the measurement vector, we refer to $\boldsymbol{\Gamma}$ as the *measurement matrix*.

Finally, during the data processing phase, the receiver maps the measurement vector obtained in the data collection phase to the sensing results, which is a vector indicating the probabilities that some objects exist in the M space grids. Given the control matrix $\boldsymbol{C}$, the mapping is modeled through a parameterized function, i.e., $\hat{\boldsymbol{p}} = \boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$ with $\boldsymbol{w}$ being the parameter vector that is referred to as the *mapping of the received signals*. Moreover, the result of the mapping, i.e., $\hat{\boldsymbol{p}}$, is an M-dimensional real-valued vector. Specifically, its m-th element, i.e., $\hat{p}_m \in [0, 1]$, indicates the probability that an object exists at the m-th space grid. Therefore, $(1 - \hat{p}_m)$ indicates the probability that the m-th space grid is empty.

III. PROBLEM FORMULATION

In this section, we formulate the optimization problem for maximizing the sensing accuracy for the considered scenario. We adopt the *cross-entropy loss* as the objective function to measure the sensing accuracy, as minimizing the cross-entropy loss function can significantly improve the accuracy of classification and prediction [31]. In other words, the sensing accuracy is inversely proportional to the cross-entropy loss.

We define the cross-entropy loss in the considered scenario as

$$L_{\mathrm{CE}} = -\mathbb{E}_{\boldsymbol{\nu} \in \mathcal{V}} \Big[\sum_{m=1}^{M} p_m(\boldsymbol{\nu}) \cdot \ln(\hat{p}_m) + (1 - p_m(\boldsymbol{\nu})) \cdot \ln(1 - \hat{p}_m) \Big], \tag{7}$$

where $\mathcal{V}$ denotes the set of all possible reflection coefficient vectors corresponding to the existence of objects in the target space, and $p_m(\boldsymbol{\nu})$ is a binary variable indicating the object existence in the m-th space grid. Specifically, $p_m(\boldsymbol{\nu})$ can be expressed as

$$p_m(\boldsymbol{\nu}) = \begin{cases} 0, & \text{if } |\nu_m| = 0, \\ 1, & \text{otherwise.} \end{cases} \tag{8}$$

---

[1] If the environment is changing rapidly, $h_{\mathrm{rl}} \cdot \sqrt{P} \cdot x$ can be considered as an additional complex Gaussian noise [30], and $\tilde{\boldsymbol{\sigma}}$ in (6) is composed of the difference of the noise signals at the Rx and that of the environmental scattering signals, and thus its variance is $2\epsilon + 2\epsilon_{\mathrm{hl}}$.

In (7), $\hat{\boldsymbol{p}}$ is determined by $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$. Generally, the parameterized function $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$ can take any form. For example, it can be a linear function, i.e., $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}}) = \boldsymbol{W}\tilde{\boldsymbol{y}} + \boldsymbol{w}'$, where $\boldsymbol{W}$ and $\boldsymbol{w}'$ are determined by $\boldsymbol{w}$ and obtained by minimizing the mean squared error of the sensing results [32]. Besides, $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$ can also be a nonlinear decision function, which determines the sensing results of $\tilde{\boldsymbol{y}}$ by using conditional probabilities [33]. In this paper, we consider that $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$ is nonlinear and is modeled as a neural network, where the elements of $\boldsymbol{w}$ stand for the weights of the connections and the biases of the nodes. We refer to the neural network for $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$ as the *sensing network*.

The optimization problem for the considered metasurface-assisted scenario that maximizes the sensing accuracy can be formulated as the following cross-entropy minimization problem, where the control matrix and the mapping of the received signals parameter are the optimization variables, i.e.,

$$\text{(P1)}: \min_{\boldsymbol{C},\boldsymbol{w}} \; L_{\text{CE}}(\boldsymbol{C}, \boldsymbol{w}), \tag{9}$$

$$s.t. \; (\hat{p}_1, \ldots, \hat{p}_M) = \boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}}), \tag{10}$$

$$\tilde{\boldsymbol{y}} = \sqrt{P} \cdot x \cdot (\boldsymbol{C} - \boldsymbol{C}_0)\boldsymbol{A} + \tilde{\boldsymbol{\sigma}}, \tag{11}$$

$$\boldsymbol{C} = (\boldsymbol{c}_1^T, \ldots, \boldsymbol{c}_K^T)^T, \tag{12}$$

$$\boldsymbol{c}_k = (\hat{\boldsymbol{o}}(c_{k,1}), \ldots, \hat{\boldsymbol{o}}(c_{k,N})), \; \forall k \in [1, K], \tag{13}$$

$$c_{k,n} \in [1, N_S], \; \forall k \in [1, K], n \in [1, N]. \tag{14}$$

In (P1), (9) indicates that the objective is to minimize the cross-entropy loss by optimizing $\boldsymbol{C}$ and $\boldsymbol{w}$. As $\hat{\boldsymbol{p}}$ is determined by $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$ and $\tilde{\boldsymbol{y}}$ is determined by the control matrix $\boldsymbol{C}$, L_{CE} defined in (7) can be expressed as a function of $\boldsymbol{C}$ and $\boldsymbol{w}$. Constraint (10) indicates that the probabilities for the M space grids to contain objects are calculated from the mapping of the received signals, i.e., $\boldsymbol{f}^{\boldsymbol{w}}(\tilde{\boldsymbol{y}})$. Constraint (11) indicates that the measurement vector is determined by the control matrix $\boldsymbol{C}$ as in (6). Besides, constraints (12)-(14) are due to the definition of the control matrix in Section II-C. Since the control matrix is a binary matrix and $\boldsymbol{w}$ is a real-valued vector, (P1) is a mixed-integer optimization problem and is NP-hard.

To tackle it efficiently, we decompose (P1) into two sub-problems, i.e., (P2), and (P3), as follows:

$$\text{(P2)}: \min_{\boldsymbol{w}} \; L_{\text{CE}}(\boldsymbol{C}, \boldsymbol{w}), \quad s.t. \text{ (10)}. \tag{15}$$

$$\text{(P3)}: \min_{\boldsymbol{C}} \; L_{\text{CE}}(\boldsymbol{C}, \boldsymbol{w}), \quad s.t. \text{ (11) to (14)}. \tag{16}$$

In (P2), we minimize the cross-entropy loss by optimizing $\boldsymbol{w}$ given $\boldsymbol{C}$, and in (P3), we minimize the cross-entropy loss by optimizing $\boldsymbol{C}$ given $\boldsymbol{w}$. Based on the alternating optimization technique [34], a locally optimal solution of (P1) can be solved by iteratively solving (P2) and (P3). Nevertheless, given $\boldsymbol{w}$, (P3) is still hard to solve due to the large number of integer variables in the control matrix. Moreover, the number of iterations for solving (P2) and (P3) can be large before converging to the local optimum of (P1). If traditional methods, such as exhaustive search and branch-and-bound algorithms, are applied, they will result in a high computational complexity. To solve (P2) and (P3) efficiently, we develop an MDP framework and solve it by proposing an PRPG algorithm, which are discussed in the next section. Furthermore, the convergence of the proposed algorithm to solve (P1) is analyzed in Section V.

IV. ALGORITHM DESIGN

In this section, we formulate an MDP framework for (P2) and (P3) in Subsection A and propose a deep reinforcement learning algorithm named PRPG to solve it in Subsection B.

A. MDP Formulation

In (P3), the optimization variable $\boldsymbol{C}$ is composed of a large number of binary variables satisfying the constraints (12)-(14), which makes (P3) an integer optimization problem which is NP-hard and difficult to solve. Nevertheless, the metasurface can be considered as an intelligent agent who determines the configuration of each reconfigurable element for each beamformer pattern sequentially, and is rewarded by the negative cross-entropy loss. In this regard, the integer optimization problem (P3) can be considered as a decision optimization problem for the metasurface, which can be solved efficiently by the deep reinforcement learning technique, since it is efficient to solve highly-complexed decision optimization problems for intelligent agents [35], [36]. As deep reinforcement learning algorithms require the target problem to be formulated as an MDP, we formulate (P2) and (P3) as an MDP, so that we can solve them by proposing an efficient deep learning algorithm.

An MDP encompasses an environment and an agent, and consists of four components: the set of states $\mathcal{S}$, the set of available actions $\mathcal{A}$, the state transition function $\mathcal{T}$, and the reward function $\mathcal{R}$ [23]. The states in $\mathcal{S}$ obey the Markov property, i.e., each state only depends on the previous state and the adopted action. Suppose the agent takes action a in state s, and the consequent state s' is given by the transition function $\mathcal{T}$, i.e., $s' = T(\boldsymbol{s}, a)$. After the state transition, the agent receives a reward that is determined by the reward function $\mathcal{R}$, i.e., $\mathcal{R}(\boldsymbol{s}', \boldsymbol{s}, a)$.

To formulate the MDP framework for (P2) and (P3), we view the metasurface as the agent, and the RF sensing scenario including the surroundings, the RF transceiver, and the objects in the target space are regarded, altogether, as the environment. We consider the state of the metasurface the current control matrix, i.e., $\boldsymbol{C}$ and the action of the metasurface as selecting the configuration of a reconfigurable element for a beamformer pattern. Thus, the actions of the metasurface determine the elements in the control matrix $\boldsymbol{C}$. Therefore, the next state of the MDP is determined by the current state and the action, and the Markov property is satisfied. In the following, we describe the components of the MDP framework in detail.

1) State: In the MDP of the metasurface-assisted RF sensing scenarios, the state of the environment is defined as

$$\boldsymbol{s} = (k, n, \boldsymbol{C}), \tag{17}$$

where $k \in [1, K]$ and $n \in [1, N]$ are the row and column indexes for the control matrix indicating the configuration that the metasurface aims to select. Besides, $\boldsymbol{C}$ in (17) denotes the current control matrix of the metasurface in state $\boldsymbol{s}$. The initial state of the MDP framework is denoted by $\boldsymbol{s}_0 = (1, 1, \boldsymbol{C}_0)$, where $\boldsymbol{C}_0$ is the control matrix of the metasurface whose reconfigurable elements are in the first/default configuration.

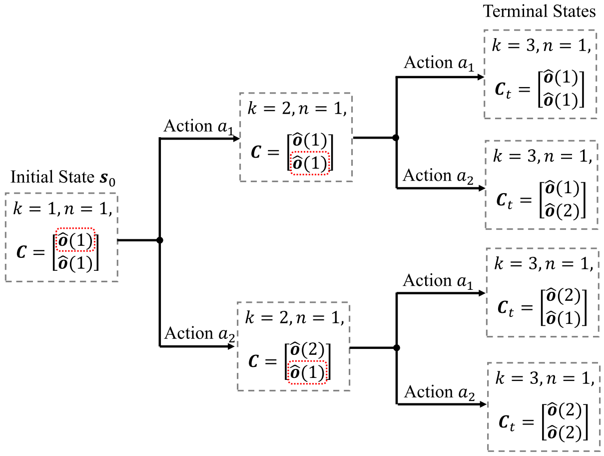

Fig. 3. Example of the state transition in the formulated MDP, with $K = 2$, $N = 1$, and $N_S = 2$.

We refer to the states with indices $(k, n) = (K+1, 1)$ as the *terminal states*. When the terminal states are reached, all the configurations in the control matrix have been selected.

2) Action: In each state $\boldsymbol{s} = (k, n, \boldsymbol{C})$, the metasurface selects the state of the n-th reconfigurable element in the k-th frame. The action set of the metasurface in each state can be expressed as $\mathcal{A} = \{1, \ldots, N_S\}$, where the j-th action ($i \in [1, N_S]$) indicates that the metasurface selects the target configuration to be the j-th configuration. In other words, the metasurface sets $(\boldsymbol{C})_{k,n} = \hat{\boldsymbol{o}}(a)$, $a \in \mathcal{A}$.

3) State Transition Function: After the metasurface selects the action, the MDP framework transits into the next state, $\boldsymbol{s}' = (k', n', \boldsymbol{C}')$, if $(k, n) \neq (K, N)$. If $(k, n) = (K, N)$ on the other hand, the agent enters the terminal state of the MDP. For the non-terminal states, the elements of state $\boldsymbol{s}'$ given $\boldsymbol{s}$ and a can be expressed as follows

$$k' = k+1, \quad n' = \mathrm{mod}(n+1, N) + 1, \tag{18}$$

$$(\boldsymbol{C}')_{k'',n''} = \begin{cases} (\boldsymbol{C})_{k'',n''}, \text{ if } (k'', n'') \neq (k, n), \\ \hat{\boldsymbol{o}}(a) \text{ if } (k'', n'') = (k, n), \end{cases} \\ \forall k'' \in [1, K],\ n'' \in [1, N]. \tag{19}$$

An example of the state transition is illustrated in Fig. 3, where $N_S = 2$, $K = 2$, and $N = 1$. In Fig. 3, the red dotted box indicates the element of $\boldsymbol{C}$ that is determined by the action in the current state. If $(k, n) = (3, 1)$, it can be observed that all the configurations of the control matrix have been determined, and the MDP transits into the terminal states, where control matrix is denoted by $\boldsymbol{C}_t$.

4) Reward Function: In general MDP frameworks, the reward is a value obtained by the agent from the environment and quantifies the degree to which the agent's objective has been achieved [23]. The reward for the agent is defined as the negative cross-entropy loss of the mapping of the received signals given the control matrix determined in the terminal states. If the terminal state has not been reached, the reward for the state transition is set to be zero. Specifically, given the parameter $\boldsymbol{w}$, the reward in state $\boldsymbol{s}$ is defined as

$$\mathcal{R}(\boldsymbol{s}|\boldsymbol{w}) = \begin{cases} -L_{\mathrm{CE}}(\boldsymbol{C}_t, \boldsymbol{w}), & \text{if } \boldsymbol{s} \text{ is a terminal state,} \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

In the formulated MDP, the metasurface aims for obtaining an optimal policy to obtain the maximum reward in the terminal states. To be specific, the policy of the agent is a mapping from the state set to the available action set, i.e., $\boldsymbol{\pi} : \mathcal{S} \to \mathcal{A}$. To define the optimal policy $\boldsymbol{\pi}^*$, we first define the *state-value function* given the policy $\boldsymbol{\pi}$ and the parameter vector $\boldsymbol{w}$, which indicates the accumulated reward of the agent via a certain state. Based on (20), the state-value function can be expressed as

$$V(\boldsymbol{s}|\boldsymbol{\pi}, \boldsymbol{w}) = \begin{cases} -L_{\mathrm{CE}}(\boldsymbol{C}, \boldsymbol{w}), & \text{if } \boldsymbol{s} \text{ is a terminal state,} \\ V(\boldsymbol{s}'|\boldsymbol{\pi}, \boldsymbol{w})|_{\boldsymbol{s}'=\mathcal{T}(\boldsymbol{s}, \boldsymbol{\pi}(\boldsymbol{s}))}, & \text{otherwise,} \end{cases} \tag{21}$$

The state-value function for $\boldsymbol{\pi}$ in state $\boldsymbol{s}$ indicates the accumulated rewards of the agent after state $\boldsymbol{s}$. Based on (21), the state-value function for the initial state can be expressed as

$$V(\boldsymbol{s}_0|\boldsymbol{\pi}, \boldsymbol{w}) = -L_{\mathrm{CE}}(\boldsymbol{C}_t^{\boldsymbol{\pi}}, \boldsymbol{w}), \tag{22}$$

where $\boldsymbol{C}_t^{\boldsymbol{\pi}}$ denotes the terminal state of the metasurface adopting policy $\boldsymbol{\pi}$.

Therefore, given the parameter vector $\boldsymbol{w}$, the optimal policy of the agent in the MDP framework is given by

$$\boldsymbol{\pi}^*(\boldsymbol{w}) = \arg\max_{\boldsymbol{\pi}} V(\boldsymbol{s}_0|\boldsymbol{\pi}, \boldsymbol{w}) \iff \arg\min_{\boldsymbol{C}} L_{\mathrm{CE}}(\boldsymbol{C}, \boldsymbol{w}). \tag{23}$$

In (23), it can be observed that finding the optimal policy of the agent in the formulated MDP framework is equivalent to solving the optimal control matrix for (P3). Besides, solving (P2) is equivalent to solving the optimal $\boldsymbol{w}$ given the policy $\boldsymbol{\pi}$.

B. Progressing Reward Policy Gradient Algorithm

To jointly solve (P2) and (P3) under the formulated MDP framework, we propose a novel PRPG algorithm. The proposed algorithm can be divided into two phase, i.e., the *action selection phase* and the *training phase*, which proceed iteratively.

1) Action Selection Process: In the proposed algorithm, the agent, i.e., the metasurface, starts from the initial state $\boldsymbol{s}_0$ and adopts the policy for selecting an action in each state until reaching the terminal state. To select the current action in each state, the metasurface uses the policy $\boldsymbol{\pi}$ that maps the current state to a probability vector. To be specific, for a given state $\boldsymbol{s}$, the policy results in an N_S-dimensional probability vector denoted by $\boldsymbol{\pi}(\boldsymbol{s}|\boldsymbol{w})$, which we refer to as the *policy function*. The i-th element of $\boldsymbol{\pi}(\boldsymbol{s}|\boldsymbol{w})$ ($i \in [1, N_S]$), i.e., $\pi_i(\boldsymbol{s}|\boldsymbol{w})$, is in the range $[0, 1]$ and denotes the probability of selecting the action a_i in state $\boldsymbol{s}$. Besides, $\boldsymbol{\pi}(\boldsymbol{s}|\boldsymbol{w})$ ($i \in [1, N_S]$) satisfies the constraint $\sum_{i=1}^{N_S} \pi_i(\boldsymbol{s}|\boldsymbol{w}) = 1$.

However, since the state contains the current control matrix that has $K \cdot N \cdot N_S$ binary variables, the agent faces a large state space, and the policy function is hard to be modeled by using simple functions. To handle this issue, we adopt a

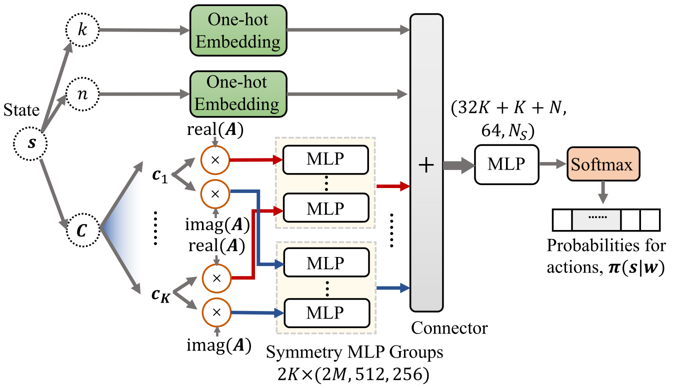

Fig. 4. Network structure of the policy network used in the proposed algorithm.

neural network to model the policy function as neural networks are a powerful tool to handle large state spaces [37]. The adopted neural network is referred to as the policy network, and we train the policy network by using the policy gradient algorithm [35]. Specifically, the policy network is denoted by $\boldsymbol{\pi}^{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{w})$, where $\boldsymbol{\theta}$ denotes the parameters of the policy network and comprises the connection weights and the biases of the activation functions in the neural network.

The structure of the policy network is shown in Fig. 4. In state $\boldsymbol{s}$, k and n are embedded as a K-dimensional and an N-dimensional vectors, respectively, where the k-th and n-th elements in the vectors are ones and the other elements are zeros. Specifically, we refer to the resulting vectors as the *one-hot* vectors. As for $\boldsymbol{C}$, since the RF sensing for the target space is determined by $\boldsymbol{C}\boldsymbol{A}$ as shown in (4), we first divide $\boldsymbol{C}$ to its real and imaginary parts and right-multiply them by the real and imaginary parts of $\boldsymbol{A}$, respectively. Then, driven by the concept of model-based learning [38], we process the result, i.e., $\boldsymbol{C}\boldsymbol{A}$, by multi-layer perceptrons (MLPs). Besides, since the K beamformer patterns are symmetric in their physical meaning and changing their order does not impact the sensing performance, the MLPs that extract feature vectors from $\boldsymbol{c}_1$ to $\boldsymbol{c}_K$ need to be symmetric. This can be achieved by utilizing two *symmetric MLP groups*, each containing K MLPs with shared parameters. This significantly reduces the number of parameters and thus facilitates the training of the policy network. The sizes of the MLPs are labeled in Fig. 4. For example, $(2M, 512, 256)$ indicates that each MLP in a symmetric group that has three layers whose sizes are $2M$, 512, and 256, respectively. Then, the one-hot vectors and the $2K$ extracted feature vectors are connected and input to the final MLP. The result of the final MLP is fed into the *softmax* layer which produces an N_S-dimensional vector indicating the probability of selecting the N_S actions.

2) Training Process: The purpose of the training process is two fold: *(a)* To make the policy network improves the current policy in the action selection based on (23). *(b)* To make the mapping of the received signals incurs lower cross-entropy loss. Accordingly, the training process consists of two parts, i.e., training the policy network and training the sensing network. In the training of the policy network, we adopt the policy gradient method [23]. Besides, between two consecutive policy network updates, the sensing network is trained, which makes the rewards for the metasurface change progressively larger. Due to these characteristics, the proposed algorithm is named as *progressing reward policy gradient* algorithm.

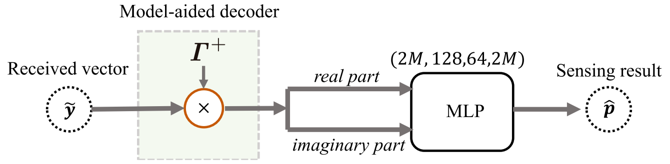

Fig. 5. Sensing network of the metasurface.

a) Training of the policy network: To collect the training data for the policy network, a *replay buffer* is adopted in order to store the experiences of the agent during the state transitions. The replay buffer of the agent is denoted by $\mathcal{B} = \{\boldsymbol{e}\}$. The stored experience in the replay buffer is given by $\boldsymbol{e} = (\boldsymbol{s}, a)$. It is worth noting that, differently from the replay buffer in traditional deep reinforcement learning algorithms [35], the experience in the replay buffer does not record the reward obtained during the state transitions. This is because the rewards are determined by the current mapping of the received signals, which changes as $\boldsymbol{w}$ being updated. Thus, we propose that the rewards are calculated when the training process is invoked, instead of being recorded in the replay buffer.

We define a *training epoch* (or epoch in short) as the state transition process from the initial state to a terminal state. The experience of the agent within an epoch is stored into the replay buffer and used for training, which is discarded after being used. Based on the policy gradient theorem [23], in the training process, the gradient of $V(\boldsymbol{s}_0|\boldsymbol{\pi},\boldsymbol{w})$ with respect to $\boldsymbol{\theta}$ satisfies

$$\nabla_{\boldsymbol{\theta}} V(\boldsymbol{s}_0|\boldsymbol{\pi},\boldsymbol{w}) \propto \mathbb{E}_{\mathcal{B},\boldsymbol{\pi}^{\boldsymbol{\theta}}}\left[V(\mathcal{T}(S_t,A_t)|\boldsymbol{\theta},\boldsymbol{w})\frac{\nabla_{\boldsymbol{\theta}}\pi^{\boldsymbol{\theta}}_{A_t}(S_t|\boldsymbol{w})}{\pi^{\boldsymbol{\theta}}_{A_t}(S_t|\boldsymbol{w})}\right], \tag{24}$$

where $(S_t, A_t) \in \mathcal{B}$ are the samples of the state and action in the replay buffer of an agent following the policy $\boldsymbol{\pi}^{\boldsymbol{\theta}}$, and $Q(S_t, A_t|\boldsymbol{\theta}, \boldsymbol{w})$ denotes the reward for the agent after selecting the action A_t in S_t and then following the policy $\boldsymbol{\pi}^{\boldsymbol{\theta}}$.

To calculate the gradient in (24), the rewards for the agent in (20) need to be calculated. If $\boldsymbol{s}$ is a terminal state, the reward $R(\boldsymbol{s}|\boldsymbol{w})$ is calculated by using Monte Carlo methods [39], i.e.,

$$\begin{aligned} R(\boldsymbol{s}|\boldsymbol{w}) = -\sum_{\boldsymbol{\nu}\in\mathcal{V}}\sum_{i=1}^{N_{\mathrm{mc}}}\Bigg(\sum_{m=1}^{M} p_m(\boldsymbol{\nu})\ln(\hat{p}_m) \\ +(1-p_m(\boldsymbol{\nu}))\ln(1-\hat{p}_m)\Bigg)\Bigg|_{\hat{\boldsymbol{p}}=\boldsymbol{f}^{\boldsymbol{w}}(\boldsymbol{\Gamma}\boldsymbol{\nu}+\tilde{\boldsymbol{\sigma}}_i)}. \end{aligned} \tag{25}$$

Otherwise, $R(\boldsymbol{s}|\boldsymbol{w}) = 0$. In (25), N_{mc} indicates the number of sampled noise vectors, and $\tilde{\boldsymbol{\sigma}}_i$ is the i-th sampled noise vector. As the rewards in the non-terminal states are zero, $V(\mathcal{T}(S_t, A_t)|\boldsymbol{\theta}, \boldsymbol{w})$ is equal to the reward at the final state for S_t, A_t, and policy $\boldsymbol{\pi}^{\boldsymbol{\theta}}$.

Specifically, in (25), $\hat{\boldsymbol{p}}$ is generated by the sensing network, which is shown in Fig. 5. The sensing network consists of two parts, i.e., the *model-aided decoder* and an MLP. Firstly, the received vector is left-multiplied by the *pseudo inverse* of $\boldsymbol{\Gamma}$, which is denoted by $\boldsymbol{\Gamma}^{+}$, which can be calculated based on [40]. According to the least-square method [32], the model-aided decoder, i.e., $\hat{\boldsymbol{\nu}} = \boldsymbol{\Gamma}^{+}\boldsymbol{y}$, is the optimal linear decoder that results in the minimum mean square error (MSE) for the actual reflection vector $\boldsymbol{\nu}$, and thus can potentially increase the sensing accuracy of the sensing network. Then, $\hat{\boldsymbol{\nu}}$ is fed into a fully-connected MLP, which reconstructs the probability vector $\hat{\boldsymbol{p}}$.

In each process, $\boldsymbol{\theta}$ is updated as follows

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \cdot \mathbb{E}_{\boldsymbol{e}\in\mathcal{B}}\left[V(\mathcal{T}(S_t, A_t)|\boldsymbol{\theta}, \boldsymbol{w})\frac{\nabla_{\boldsymbol{\theta}}\pi^{\boldsymbol{\theta}}_{A_t}(S_t|\boldsymbol{w})}{\pi^{\boldsymbol{\theta}}_{A_t}(S_t|\boldsymbol{w})}\right], \tag{26}$$

where the gradient $\nabla_{\boldsymbol{\theta}}\pi^{\boldsymbol{\theta}}_{A_t}(S_t|\boldsymbol{w})$ is calculated by using the back-propagation algorithm [41], and α denotes the training rate.

b) Training of the sensing network: After updating $\boldsymbol{\theta}$, the training of the sensing network is executed. The calculated rewards from (25) are used to train the sensing network which reduces the cross-entropy loss. To be specific, the loss function used to train the sensing network can be expressed as follows, which is in accordance with the objective function in the optimization problem (P2), i.e.,

$$\mathcal{L}_I(\boldsymbol{w}) = \mathbb{E}_{(\boldsymbol{s},a)\in\mathcal{B}}[R(\boldsymbol{s}|\boldsymbol{w})]. \tag{27}$$

In each training process, $\boldsymbol{w}$ is updated by

$$\boldsymbol{w} = \boldsymbol{w} + \alpha\nabla_{\boldsymbol{w}}\mathcal{L}_I(\boldsymbol{w}), \tag{28}$$

where the gradient $\nabla_{\boldsymbol{w}}\mathcal{L}_I(\boldsymbol{w})$ is calculated by using the back-propagation algorithm.

In summary, the proposed PRPG algorithm is summarized in Algorithm 1.

Remark: Using the proposed deep reinforcement learning technique enables our proposed algorithm to handle the complicated scenarios where multiple metasurfaces are deployed. Specifically, when multiple metasurfaces are on the same plane, they can be considered as a whole, and thus the channel model in (1) needs no changes. When the multiple metasurfaces are on different planes, the channel model needs to be modified to adapt to the correlation between different metasurfaces, which is left for future work. Nevertheless, since the problem formulation and the proposed algorithm are independent of the specific channel model, the proposed problem formulation and algorithm can also be applied to general RF sensing scenarios with multiple metasurfaces.

V. ALGORITHM ANALYSIS

In this section, we analyze the computational complexity and the convergence of the proposed algorithm in Subsections A and B, respectively. In addition, in Subsection C, we derive a non-trivial lower-bound for the sensing accuracy based on an upper-bound for the cross-entropy loss given a control matrix.

Algorithm 1 Proposed PRPG Algorithm

```
Input: Random initial network parameter vectors $\boldsymbol{\theta}$ and $\boldsymbol{w}$;
    Empty replay buffer $\mathcal{B} = \emptyset$;
    Maximum number of training epochs $N_{\mathrm{ep}}$;
    Set of reflection coefficient vectors $\mathcal{V}$;
    Number of Monte Carlo samples for noise $N_{\mathrm{mc}}$;
    Initial learning rate $\alpha_0$;
    Maximum number of training epochs $N_{\mathrm{ep}}$
Output: Optimized sensing network parameter vector $\boldsymbol{w}^*$ and the optimized policy network parameter $\boldsymbol{\theta}^*$.
for $n_{\mathrm{ep}} = 1$ to $N_{\mathrm{ep}}$ do
    Set the current state to be the initial state, i.e., $\boldsymbol{s} = \boldsymbol{s}_0$;
    # Action selection phase
    while $\boldsymbol{s}$ is not a terminal state do
        Select the configuration of the $n$-th reconfigurable element in the $k$-th frame following the probability distribution given by $\boldsymbol{\pi}^{\boldsymbol{\theta}}(\boldsymbol{s}|\boldsymbol{w})$.
        Set action $a$ as the selected configuration, and enter into the transited state $\boldsymbol{s}' = \mathcal{T}(\boldsymbol{s}, a)$;
        Store experience $\boldsymbol{e} = (\boldsymbol{s}, a)$ into replay buffer $\mathcal{B}$;
    # Training phase
    Collect all the experiences from $\mathcal{B}$, and calculate the reward for each sampled experience by using (25);
    Update parameter $\boldsymbol{\theta}$ and $\boldsymbol{w}$ by (26) and (28), respectively, where the learning rate $\alpha = \frac{\alpha_0}{1+n_{\mathrm{ep}}\cdot 10^{-3}}$;
```

A. Computational Complexity

Since the PRPG algorithm consists of two main phases, i.e., the action selection phase and the training phase, we analyze their respective computational complexities. The computational complexities are analyzed with regard to the number of beamformer patterns, K, the number of reconfigurable elements, N, the number of available configuration, N_S, and the number of space grids, M.

1) Complexity of the Action Selection Phase: In the proposed algorithm, the computationally most expensive part is the estimation of the action probabilities of the policy network. For each action selection phase, the computational complexity is given in Theorem 1.

Theorem 1 (Computational Complexity of the Action Selection Phase): In the PRPG algorithm, for the agent in each state, the complexity to calculate the action probabilities and determine the action is $\mathcal{O}(KNN_SM)$.

Proof: See Appendix I. ■

2) Complexity of the Training Process: The computational complexity of (25) is provided in Lemma 1.

Lemma 1: The computational complexity of the reward calculation in (25) is $\mathcal{O}\big(KNN_SM + K^2M + M^2\big)$.

Proof: See Appendix II ■

The computational complexities of training the policy network and the sensing network are given in Lemma 2.

Lemma 2: After calculating the rewards, the complexity of training the sensing network and the policy network are $\mathcal{O}(M^2)$ and $\mathcal{O}(N_S(K + N + M))$, respectively. If a single

MLP is used to substitute the symmetric MLP group, the computational complexity of training the policy network is $\mathcal{O}(KMN_S + NN_S)$.

Proof: See Appendix III. ■

It can be observed from Lemma 2 that using a symmetric MLP group instead of a single large MLP in the policy network can reduce the complexity of the training process.

Based on Lemmas 1 and 2, the total computational complexity of each training process is provided in Theorem 2.

Theorem 2: (Computational Complexity of the Training Process) The computational complexity of each training phase of the PRPG algorithm is $\mathcal{O}\big(KNN_SM + K^2M + M^2\big)$.

Proof: See Appendix IV. ■

B. Convergence Analysis

The detailed convergence analysis of the PRPG algorithm is based on the convergence analysis of the block stochastic gradient (BSG) algorithm. We denote $\boldsymbol{w}$ by $\boldsymbol{x}_1$ and denote $\boldsymbol{\theta}$ by $\boldsymbol{x}_2$, and thus the objective function in (P1) can be denoted by $F(\boldsymbol{x}_1, \boldsymbol{x}_2) = L_{\mathrm{CE}}(\boldsymbol{C}_t^{\boldsymbol{\pi}^{\boldsymbol{\theta}}}, \boldsymbol{w})$, where $\boldsymbol{C}_t^{\boldsymbol{\pi}^{\boldsymbol{\theta}}}$ indicates the control matrix in the terminal state for the metasurface with policy $\boldsymbol{\pi}^{\boldsymbol{\theta}}$. Based on [42], a BSG algorithm for solving (P1) is formulated as Algorithm 2, whose convergence analysis is given by Lemma 3.

Lemma 3: Algorithm 2 converges to a locally optimal $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$ as the number of iterations $N_{\mathrm{itr}} \to \infty$, given that the following conditions are satisfied:

1) There exist a constant c and a constant ε such that, for each iteration indexed by j, the inequalities $\|\mathbb{E}[\tilde{\boldsymbol{g}}_i^j - \nabla_{\boldsymbol{x}_i} F(\boldsymbol{x}_1, \boldsymbol{x}_2)]\|_2 \le c \cdot \max_i(\alpha_i^j)$ and $\mathbb{E}[\|\tilde{\boldsymbol{g}}_i^j - \nabla_{\boldsymbol{x}_i} F(\boldsymbol{x}_1, \boldsymbol{x}_2)\|^2] \le \varepsilon^2$, $i = 1, 2$ are fulfilled.
2) There exists a uniform Lipschitz constant $\varrho > 0$ such that
$$\sum_{i=1,2} \|\nabla_{\boldsymbol{x}_i} F(\boldsymbol{x}_1, \boldsymbol{x}_2) - \nabla_{\boldsymbol{x}_i} F(\boldsymbol{x}_1', \boldsymbol{x}_2')\|_2^2 \le \varrho^2 \sum_{i=1,2} \|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2^2.$$
3) There exists a constant ψ such that $\mathbb{E}[\|\boldsymbol{x}_1^j\|_2^2 + \|\boldsymbol{x}_2^j\|_2^2] \le \psi^2$, $\forall j$.

Proof: Please refer to Corollary 2.12 in [42], where the assumptions required in Corollary 2.12 in [42] are equivalent to the three conditions in Lemma 3. ■

Comparing Algorithms 1 and 2, we can observe that the only difference between the two algorithms is in the functions for updating the parameters. Nevertheless, solving the minimization problem (30), we can derive that (30) is equivalent to

$$\boldsymbol{x}_i^j = \boldsymbol{x}_i^{j-1} - \alpha_i^j \tilde{\boldsymbol{g}}_i^j. \tag{29}$$

As the learning rate sequence $\{\alpha_i^j\}_j$ in Algorithm 2 can be arbitrarily selected, the parameter update of Algorithms 1 and 2 are essentially equivalent. In this regard, the proposed PRPG algorithm can be categorized as an BSG algorithm, whose convergence analysis follows Lemma 3.

Algorithm 2 BSG Algorithm for Solving (P1)

```
Input: Starting point $x_i^0$, $i = 1, 2$;
       Learning rate sequence $\{\alpha_i^j; i = 1, 2\}_{j=1,2,\ldots}$;
       Maximum number of iterations $N_{\mathrm{itr}}$;
       Monte Carlo sampling size of the random noise $N_{\mathrm{mc}}$.
Output: Optimized $x_1^*$ and $x_2^*$ for (P1).
for $j = 1, 2, \ldots, N_{\mathrm{itr}}$ do
    for $i = 1, 2$ do
        Compute sample gradient for the $w$ in the $j$-th iteration by $\tilde{g}_i^j = \nabla_{x_i} F(x_{<i}^j, x_{\ge i}^{(j-1)})$
        Update parameter $x_i$ by
        $x_i^j = \arg\min_{x_i} (\tilde{g}_i^j)^T (x_i - x_i^{j-1}) + \frac{1}{2\alpha_i^j} \|x_i - x_i^{j-1}\|_2^2.$
Output $(x_1^{N_{\mathrm{itr}}}, x_2^{N_{\mathrm{itr}}})$ as $(x_1^*, x_2^*)$;
```

However, since neural networks are used in the mapping of the received signals and the policy function, the conditions in Lemma 3 are hard to be proven theoretically. Therefore, in addition to the theoretical analyses provided above, we also analyze the convergence through practical simulations in Section VI.

Moreover, the obtained solution by the proposed deep learning algorithm is a locally optimal solution of (P1). As shown in Algorithm 1, we iteratively solve (P2) and (P3) by updating $\boldsymbol{\theta}$ using (26) and updating $\boldsymbol{w}$ using (28), respectively. Based on the Q-learning algorithm [23], updating $\boldsymbol{\theta}$ with the aim to maximize the total reward is equivalent to finding $\boldsymbol{C}$ minimizing L_{CE} given $\boldsymbol{w}$. Besides, it can be observed that updating $\boldsymbol{w}$ directly minimizes L_{CE} given $\boldsymbol{C}$. When the iteration terminates, updating the variables $\boldsymbol{C}$ or $\boldsymbol{w}$ will not lead to a lower objective function value, i.e., the cross-entropy loss. Therefore, the solution obtained by the proposed Algorithm 1 is a locally optimal solution of the original problem (P1).

C. Lower Bound for Sensing Accuracy

In this section, we compute a lower-bound for the sensing accuracy in (P2) given the control matrix $\boldsymbol{C}$. To derive a lower-bound, we assume that the received RF signals are mapped to the sensing results by using an optimal linear decoder and a threshold judging process. In the following, we first provide the detection criterion for sensing, and then derive a lower-bound for the sensing accuracy by leveraging an upper-bound for the cross-entropy loss.

1) Detection Criterion for Sensing: The reconstructed reflection coefficient vector from the linear decoder can be expressed as

$$\hat{\boldsymbol{\nu}} = \boldsymbol{\Gamma}^{+} \tilde{\boldsymbol{y}} = \boldsymbol{\Gamma}^{+} \boldsymbol{\Gamma} \boldsymbol{\nu} + \boldsymbol{\Gamma}^{+} \tilde{\boldsymbol{\sigma}}. \tag{30}$$

Based (30), we analyze the probability distribution of the random variable $\hat{\nu}_m$, i.e., the m-th element of $\hat{\boldsymbol{\nu}}$. We denote the m-th row vectors of $\boldsymbol{\Gamma}^{+}$ and $\boldsymbol{\Gamma}^{+}\boldsymbol{\Gamma}$ as $\boldsymbol{\gamma}_m$ and $\boldsymbol{\xi}_m$, respectively. Then, $\hat{\nu}_m = \boldsymbol{\xi}_m \boldsymbol{\nu} + \boldsymbol{\gamma}_m \tilde{\boldsymbol{\sigma}}$. The emptiness of the space grids other than the m-th space grid is modeled by the vector $\boldsymbol{q}_{-m}$, where $q_{-m,m'} = 0$ and 1 indicate that the m' space grid is empty and nonempty, respectively, ($m' \in [1, M]$, $m' \ne m$). When the m-th space grid is empty (or nonempty), we denote the probability density functions (PDFs) of the real and imaginary parts of $\hat{\nu}_m$, i.e., $\hat{\nu}_{R,m}$ and $\hat{\nu}_{I,m}$, by $\mathcal{P}^0_{R,i}(x)$ and $\mathcal{P}^0_{I,i}(x)$ (or $\mathcal{P}^1_{R,i}(x)$ and $\mathcal{P}^1_{I,i}(x)$), respectively.

We judge the emptiness of the m-th space grid according to the sum of $\hat{\nu}_{R,m}$ and $\hat{\nu}_{I,m}$, i.e., $\mu_m = \hat{\nu}_{R,m} + \hat{\nu}_{I,m}$. When the m-th space grid is empty, given $\boldsymbol{q}_{-m}$, the sum of $\hat{\nu}_{R,m}$ and $\hat{\nu}_{I,m}$, i.e., μ_m, follows a normal distribution, i.e., $\mu_m \sim \mathcal{N}(0, \epsilon^0_m(\boldsymbol{q}_{-m}))$, where

$$\epsilon^0_m(\boldsymbol{q}_{-m}) = \sum_{\substack{m' \neq m,\\ m' \in \mathcal{M}}} q_{-m,m'} \cdot \epsilon_{\mathrm{ref},m'} \cdot (\|\boldsymbol{\xi}_{R,m'}\|^2 + \|\boldsymbol{\xi}_{I,m'}\|^2) + \sum_{m' \in \mathcal{M}} \epsilon \cdot (\|\boldsymbol{\gamma}_{R,m'}\|^2 + \|\boldsymbol{\gamma}_{I,m'}\|^2). \tag{31}$$

where $\mathcal{M}$ is the set of indexes of M space grids, and the subscripts R and I indicate the real and imaginary parts of a vector, respectively. The first summand in (31) corresponds to the variance due to the reflection coefficients at the space grids other than the m-th space grid, and the second addend in (31) corresponds to the variance due to the noise at the Rx unit.

On the other hand, when the q-th space grid is nonempty, the variance due to reflection coefficient of the m-th space grid needs to be added. Denoting the variance of the reflection coefficient of the m-th space grid by $\epsilon_{\mathrm{ref},m}$, we can express the variance of μ_m as

$$\epsilon^1_m(\boldsymbol{q}_{-m}) = \epsilon^0_m(\boldsymbol{q}_{-m}) + \epsilon_{\mathrm{ref},m} \cdot (\|\boldsymbol{\xi}_{R,m}\|^2 + \|\boldsymbol{\xi}_{I,m}\|^2). \tag{32}$$

Given the emptiness of the m-th space grid, the PDF of μ_m can be written as follows

$$\mathcal{P}^i_m(x) = \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} P_m(\boldsymbol{q}_{-m}) \mathcal{P}_{norm}(x; 0, \epsilon^i_m(\boldsymbol{q}_{-m})),\ i = 0, 1 \tag{33}$$

where $\mathcal{Q}_{-m}$ indicates the set of all possible $\boldsymbol{q}_{-m}$, $\mathcal{P}_{norm}(x; 0, \epsilon^i_m(\boldsymbol{q}_{-m}))$ $(i = 0, 1)$ denotes the PDF of a normal distribution with zero mean and variance $\epsilon^i_m(\boldsymbol{q}_{-m})$, and $P_m(\boldsymbol{q}_{-m})$ denotes the probability for the existence indicated by $\boldsymbol{q}_{-m}$ to be true, i.e.,

$$P_m(\boldsymbol{q}_{-m}) = \prod_{m' \neq m, m' \in \mathcal{M}} Pr_{m'}(q_{-m,m'}), \tag{34}$$

where $Pr_{m'}(x)$ with x being 0 and 1 indicates the probabilities that the m'-th space grid are empty and nonempty, respectively.

We use the difference between $\mathcal{P}^1_m(\boldsymbol{q}_{-m})$ and $\mathcal{P}^0_m(\boldsymbol{q}_{-m})$ as the *judgement variable* to determine whether the m-th space grid is empty or not. To facilitate the analysis, we adopt the *log-sum* as a substitute for the sum in (33). Therefore, the judgement variable can be calculated as

$$\tau_m = \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} \ln\left(p_m(\boldsymbol{q}_{-m}) \mathcal{P}_{norm}(x; 0, \epsilon^1_m(\boldsymbol{q}_{-m}))\right) - \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} \ln\left(p_m(\boldsymbol{q}_{-m}) \mathcal{P}_{norm}(x; 0, \epsilon^0_m(\boldsymbol{q}_{-m}))\right). \tag{35}$$

It can be observed from (35) that τ_m increases as $\mathcal{P}^1_m(\mu_m)$ increases, and that it decreases as $\mathcal{P}^0_m(\mu_m)$ increases. Therefore, we can judge the emptiness of the m-th space grid through the value of τ_m. Specifically, the sensing result of the m-th space grid is determined by comparing the judging variable τ_m with the *judging threshold*, which is denoted by ρ_m. If $\tau_m \leq \rho_m$, the sensing result of the m-th space grid is "empty", which is denoted by the hypothesis $\mathcal{H}_0$. Otherwise, if $\tau_m > \rho_m$, the sensing result is "non-empty", which is denoted by the hypothesis $\mathcal{H}_1$. After simplifying (35), the *detection criterion* for $\mathcal{H}_0$ and $\mathcal{H}_1$ can be expressed as

$$\tau_m = \mu^2_m \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} \frac{\epsilon^1_m(\boldsymbol{q}_{-m}) - \epsilon^0_m(\boldsymbol{q}_{-m})}{2\epsilon^1_m(\boldsymbol{q}_{-m})\epsilon^0_m(\boldsymbol{q}_{-m})} - \frac{1}{2} \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} \ln\left(\frac{\epsilon^1_m(\boldsymbol{q}_{-m})}{\epsilon^0_m(\boldsymbol{q}_{-m})}\right) \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\lessgtr}} \rho_m. \tag{36}$$

Since $\mu^2_m > 0$, the range of ρ_m can be expressed $[-\frac{1}{2}\sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} \ln(\frac{\epsilon^1_m(\boldsymbol{q}_{-m})}{\epsilon^0_m(\boldsymbol{q}_{-m})}), \infty]$.

2) Upper Bound of Cross Entropy Loss: We analyze the cross-entropy loss incurred by the detection criterion in (36), which can be considered as a non-trivial upper-bound for the cross-entropy loss defined in (7). As the sensing result given by (36) is either 0 or 1, if the sensing result is accurate, the incurred cross-entropy loss will be $-\ln(1) = 0$; otherwise, the incurred cross-entropy loss will be $-\ln(0) \to \infty$. In practice, the cross-entropy loss due to an inaccurate sensing result is bounded by a large number $C_{\mathrm{In}0}$. Given $\mathcal{H}_0$ (or $\mathcal{H}_1$) being true, the probability for the sensing result to be inaccurate is the probability of $\tau_m > \rho_m$, i.e., $\Pr\{\tau_m > \rho_m | \mathcal{H}_0\}$ (or $\Pr\{\tau_m \leq \rho_m | \mathcal{H}_1\}$). Denote the probability for an object to be at the m-th space grid by $\tilde{p}_m$, and the cross-entropy loss of the m-th space grid can be calculated as

$$L_m = C_{\mathrm{In}0} \cdot (1 - \tilde{p}_m) \cdot \Pr\{\tau_m > \rho_m | \mathcal{H}_0\} + C_{\mathrm{In}0} \cdot \tilde{p}_m \cdot \Pr\{\tau_m \leq \rho_m | \mathcal{H}_1\}, \tag{37}$$

where $\Pr\{\tau_m > \rho_m | \mathcal{H}_0\}$ and $\Pr\{\tau_m \leq \rho_m | \mathcal{H}_1\}$ can be calculated by using Proposition 1.

Proposition 1: The conditional probability for sensing the m-th space grid inaccurately can be calculated as follows

$$\begin{aligned}\Pr\{\tau_m > \rho_m | \mathcal{H}_0\} &= \Pr\{\mu^2_m > \hat{\rho}_m | \mathcal{H}_0\}\\ &= 1 - \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} P_m(\boldsymbol{q}_{-m}) \cdot \mathrm{erf}\left(\sqrt{\frac{\hat{\rho}_m}{2\epsilon^0_m(\boldsymbol{q}_{-m})}}\right),\end{aligned} \tag{38}$$

$$\begin{aligned}\Pr\{\tau_m \leq \rho_m | \mathcal{H}_1\} &= \Pr\{\mu^2_m \leq \hat{\rho}_m | \mathcal{H}_1\}\\ &= \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} P_m(\boldsymbol{q}_{-m}) \cdot \mathrm{erf}\left(\sqrt{\frac{\hat{\rho}_m}{2\epsilon^1_m(\boldsymbol{q}_{-m})}}\right),\end{aligned} \tag{39}$$

where $\mathrm{erf}(\cdot)$ denotes the *error function* [33], and

$$\hat{\rho}_m = \frac{\frac{1}{2}\sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} \ln(\epsilon^1_m(\boldsymbol{q}_{-m})/\epsilon^0_m(\boldsymbol{q}_{-m})) + \rho_m}{\sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} \frac{\epsilon^1_m(\boldsymbol{q}_{-m}) - \epsilon^0_m(\boldsymbol{q}_{-m})}{\epsilon^1_m(\boldsymbol{q}_{-m}) \cdot \epsilon^0_m(\boldsymbol{q}_{-m})}}. \tag{40}$$

Proof: Based on (36), the judging condition $\tau_m \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\lessgtr}} \rho_m$ is equivalent to $\mu_m^2 \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\lessgtr}} \hat{\rho}_m$. Therefore, $\Pr\{\mu_m^2 > \hat{\rho}_m|\mathcal{H}_0\} = \Pr\{\tau_m > \rho_m|\mathcal{H}_0\}$ and $\Pr\{\mu_m^2 \leq \hat{\rho}_m|\mathcal{H}_1\} = \Pr\{\tau_m \leq \rho_m|\mathcal{H}_1\}$. Also, given $\boldsymbol{q}_{-m}$, μ_m^2 follows a chi-squared distribution with one degree of freedom. Therefore, the cumulative distribution function of μ_m^2 is a weighted sum of error functions, and thus the conditional probabilities can be calculated by using (38) and (39). ∎

Besides, we can observe in (37) that L_m is determined by the judgment threshold ρ_m. Then, based on (37) to (40), $\partial L_m/\partial \rho_m$ can be calculated as

$$\partial L_m/\partial \rho_m = -\frac{2C_{\mathrm{In}0}}{\sqrt{\pi}} \cdot \frac{\partial \hat{\rho}_m}{\partial \rho_m} \cdot \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}} P_m(\boldsymbol{q}_{-m}) \cdot \phi_m(\boldsymbol{q}_{-m}), \quad (41)$$

$$\phi_m(\boldsymbol{q}_{-m}) = \frac{(1-\tilde{p}_m) \cdot e^{-\hat{\rho}_m/2\epsilon_m^0(\boldsymbol{q}_{-m})}}{\sqrt{8\epsilon_m^0(\boldsymbol{q}_{-m})\hat{\rho}_m}} - \frac{\tilde{p}_m \cdot e^{-\hat{\rho}_m/2\epsilon_m^1(\boldsymbol{q}_{-m})}}{\sqrt{8\epsilon_m^1(\boldsymbol{q}_{-m})\hat{\rho}_m}}. \quad (42)$$

Then, the optimal ρ_m^* can be obtained by solving $\partial L_m/\partial \rho_m = 0$. Denoting the minimal L_m corresponding to ρ_m^* as L_m^*, the upper-bound for the cross-entropy loss in (7) can be calculated as

$$L_{\mathrm{ub}} = \sum_{m \in \mathcal{M}} L_m^*. \quad (43)$$

When the emptiness of the space grids other than the m-th is given, the upper-bound of the cross-entropy loss can be calculated from Proposition 2. Since the sensing accuracy is inversely proportional to the cross-entropy loss, a lower-bound for the sensing accuracy is derived.

Proposition 2: When the emptiness of the space grids other than the m-th is given, i.e., $\mathcal{Q}_{-m} = \{\boldsymbol{q}_{-m}\}$, the optimal judging threshold for the m-th space grid in (44), as shown at the bottom of the next page.

Proof: The sign of $\partial L_m/\partial p_m$ is determined by $\phi_m(\boldsymbol{q}_{-m})$. We calculate the ratio between the two terms of $\phi_m(\boldsymbol{q}_{-m})$, which can be expressed as

$$\iota_m(\boldsymbol{q}_{-m}) = \frac{1-\tilde{p}_m}{\tilde{p}_m} \cdot \sqrt{\frac{\epsilon_m^1(\boldsymbol{q}_{-m})}{\epsilon_m^0(\boldsymbol{q}_{-m})}} \cdot e^{-\hat{\rho}_m \cdot \frac{\epsilon_m^1(\boldsymbol{q}_{-m})-\epsilon_m^0(\boldsymbol{q}_{-m})}{2\epsilon_m^0(\boldsymbol{q}_{-m})\cdot\epsilon_m^1(\boldsymbol{q}_{-m})}}. \quad (45)$$

Since $\epsilon_m^1(\boldsymbol{q}_{-m}) > \epsilon_m^0(\boldsymbol{q}_{-m})$ and $\hat{\rho}_m \propto \rho_m$, $\iota_m(\boldsymbol{q}_{-m})$ is a monotonic decreasing function with respect to ρ_m and $\iota_m(\boldsymbol{q}_{-m}) \geq 0$. Also, $\phi_m(\boldsymbol{q}_{-m}) \geq 0 \iff \iota_m(\boldsymbol{q}_{-m}) \geq 1$, and thus, $\partial L_m/\partial p_m \geq 0$ if and only if $\iota_m(\boldsymbol{q}_{-m}) \geq 1$. Therefore, the minimal H_m is obtained when ρ_m satisfies the condition $\iota_m(\boldsymbol{q}_{-m}) = 1$. Then, we can prove Proposition 2 by solving $\iota_m(\boldsymbol{q}_{-m}) = 1$ and considering that $\rho_m \geq -\frac{1}{2}\ln(\frac{\epsilon_m^1(\boldsymbol{q}_{-m})}{\epsilon_m^0(\boldsymbol{q}_{-m})})$. ∎

However, since the number of possible $\boldsymbol{q}_{-m}$ can be large, (typically, $|\mathcal{Q}_{-m}| = 2^{M-1}$), calculating the exact $\partial L_m/\partial \rho_m$ in (41) is time-consuming, which makes it hard to find the exact ρ_m^* and L_m^*. Therefore, in practice, we approximate H_{ub} by using a random sampled subset of $\mathcal{Q}_{-m}^{\mathrm{sam}}$, which is denoted by $\mathcal{Q}_{-m}^{\mathrm{sam}} \subset \mathcal{Q}_{-m}$.

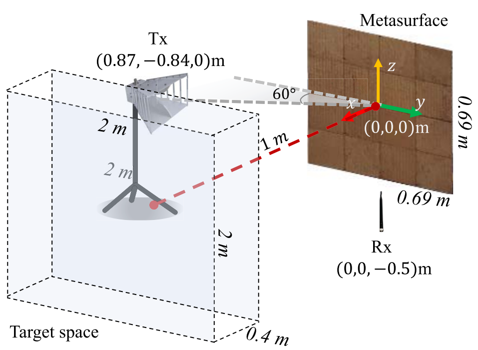

Fig. 6. Simulation layout.

Moreover, since the sign of $\partial L_m/\partial \rho_m$ is determined by the sum of $\phi_m(\boldsymbol{q}_{-m})$, and $\phi_m(\boldsymbol{q}_{-m})$ has a zero point, which can be calculated by (44). If ρ_m is less than the zero point of $\phi_m(\boldsymbol{q}_{-m})$, $\phi_m(\boldsymbol{q}_{-m}) \geq 0$; and otherwise $\phi_m(\boldsymbol{q}_{-m}) < 0$. Therefore, we use the mean of the optimal $\rho_m^*(\boldsymbol{q}_{-m})$ for each $\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}^{\mathrm{sam}}$ to estimate ρ_m^*, and approximate the upper-bound accordingly. The estimated ρ_m^* is denoted by $\tilde{\rho}_m^*$, which can be formulated as follows

$$\tilde{\rho}_m^* = \frac{1}{|\mathcal{Q}_{-m}^{\mathrm{sam}}|} \sum_{\boldsymbol{q}_{-m} \in \mathcal{Q}_{-m}^{\mathrm{sam}}} \rho_m^*(\boldsymbol{q}_{-m}), \quad (46)$$

where $\rho_m^*(\boldsymbol{q}_{-m})$ can be obtained by Proposition 2. When $|\mathcal{Q}_{-m}^{\mathrm{sam}}|$ is large enough, $\tilde{\rho}_m^*$ in (46) can approximate ρ_m^*.

Finally, given the approximated upper-bound of the cross-entropy loss as $\tilde{L}_{\mathrm{ub}}$, it can be observed from (37) that the upper-bound of average probability of sensing error for a space grid is $P_{\mathrm{err,ub}} = \tilde{L}_{\mathrm{ub}}/C_{\mathrm{In}0}$. Therefore, the lower-bound of the average sensing accuracy for a space grid is $P_{\mathrm{acc,lb}} = 1 - P_{\mathrm{err,ub}}$.

VI. SIMULATIONS AND EVALUATION

In this section, we first describe the setting of the simulation scenario and summarize the simulation parameters. Then, we provide simulation results to verify the effectiveness of the proposed PRPG algorithm. Finally, using the proposed algorithm, we evaluate the cross-entropy loss of the metasurface-assisted RF sensing scenario with respect to different numbers of sizes of the metasurface, and numbers of space grids. Besides, we also compare the proposed method with the benchmark, i.e., the MIMO RF sensing systems.

A. Simulation Settings

The layout of the considered scenario is illustrated in Fig. 6. The metasurface adopted in this paper is the same as the one used in [27], and the reflection coefficients of the reconfigurable element in different configurations are simulated by using the CST software, Microwave Studio, Transient Simulation Package [43], by assuming 60° incident RF signals with vertical polarization. Besides, to increase the reflected signal power in the simulation, we combine G reconfigurable

TABLE I
SIMULATION PARAMETERS

Parameter	**Value**
Tx antenna gain (g_T)	15.0 dBi
Rx antenna gain (g_R)	6.0 dBi
Tx power (P)	100 mW
Number of reconfigurable elements per group (G)	144
Signal frequency (f_c)	3.198 GHz
Number of available states (N_S)	4
Number of frames (K)	10
Probability of space grid being nonempty ($p_{m,1}$)	0.5
Size of reflection vector set ($\|\mathcal{V}\|$)	100
Number of space grids (M)	18
Size of random sampled subset (Q_{-m}^{sam})	1000
Power of noise (ϵ)	10^{-9} dBm
Size of space of interest (l_x, l_y, l_z)	(0.1, 0.1, 0.1) m
Initial learning rate (α_0)	0.001
Variance of reflection coefficient (ϵ_{ref})	1

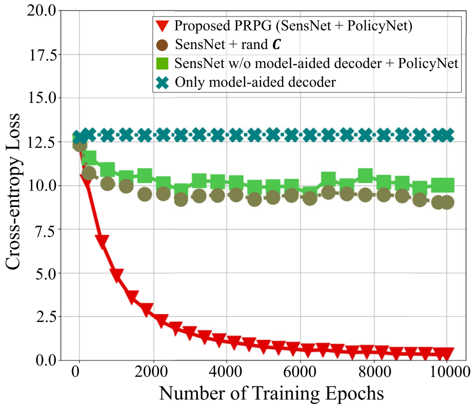

Fig. 7. Cross-entropy loss versus the number of training epochs for different algorithms.

elements as an independently controllable group. The reconfigurable elements of an independently controllable group are in the same configuration, and thus they can be considered as a single element. Therefore, the proposed algorithm is suitable for this case. The number of independently controllable group is denoted by N_G.

The origin of the coordinate is at the center of the metasurface, and the metasurface is in the y-z plane. In addition, the z-axis is vertical to the ground and pointing upwards, and the x- and y-axes are parallel to the ground. The Tx and Rx antennas are located at $(0.87, -0.84, 0)$ m and $(0, 0, -0.5)$ m, respectively. The target space is a cuboid region located at 1 m from the metasurface, and is divided into M space blocks each with size $0.1 \times 0.1 \times 0.1$ m^3. The simulation parameters are summarized in Table I.

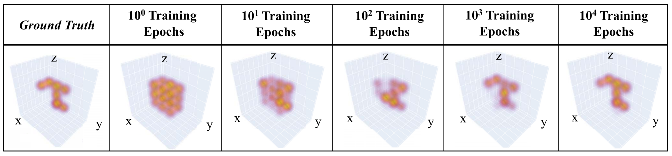

Fig. 8. Illustrations of ground-truth and the sensing results of different training epochs for a target object.

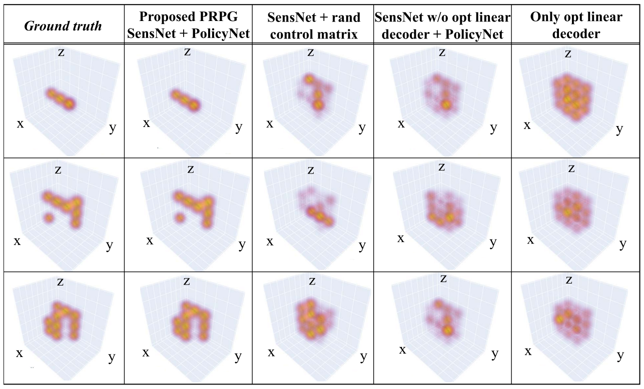

Fig. 9. Illustrations of the ground-truth and the sensing results of objects with different shapes for different algorithms.

B. Results

In Fig. 7, we compare the training results for different algorithms. Specifically, the first algorithm in the legend is the proposed PRPG algorithm where a sensing network (SensNet) and a policy network (PolicyNet) are adopted. The second algorithm adopts a sensing network but adopt a random control matrix. The third algorithm adopts both a sensing network and a policy network, but the sensing network does not contain a model-aided decoder as in the proposed algorithm. The fourth algorithm only uses the model-aided decoder to map the received signals to the sensing results.

It can be observed that the proposed PRPG algorithm converges with high speed and it results in the lowest cross-entropy loss among all the considered algorithms. In particular, Fig. 8 shows a ground-truth object and the corresponding sensing results versus the number of training epochs. As the number of training epochs increases, the sensing result approaches the ground truth and becomes approximately the same after 10^4 training epochs.

In Fig. 9, it shows the ground-truth and the sensing results for different algorithms and the target objects with different shapes. Comparing the sensing results with the ground truth, we can observe that the proposed algorithm outperforms other benchmark algorithms to a large extent. Besides, by comparing the sensing results of the proposed algorithm in the second column with the ground truth in the first column, we can observe that the proposed algorithm obtains the accurate sensing results despite the different shapes of the target objects.

$$\rho_m^*(\boldsymbol{q}_{-m}) = \begin{cases} \frac{1}{2}\ln(\frac{\epsilon_m^0(\boldsymbol{q}_{-m})}{\epsilon_m^1(\boldsymbol{q}_{-m})}), & \text{if } \tilde{p}_m > \sqrt{\frac{\epsilon_m^1(\boldsymbol{q}_{-m})}{\epsilon_m^0(\boldsymbol{q}_{-m})+\epsilon_m^1(\boldsymbol{q}_{-m})}}, \\ 2\ln(\frac{1-\tilde{p}_m}{\tilde{p}_m}) - \frac{1}{2}\ln(\frac{\epsilon_m^0(\boldsymbol{q}_{-m})}{\epsilon_m^1(\boldsymbol{q}_{-m})}), & \text{otherwise.} \end{cases} \tag{44}$$

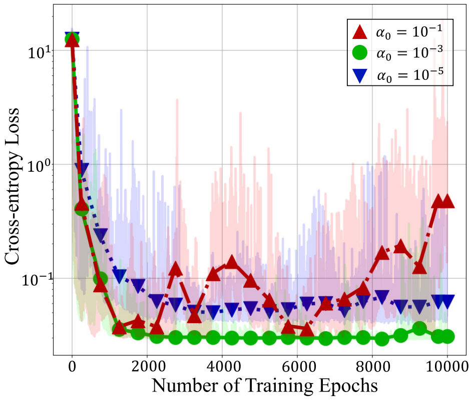

Fig. 10. Cross-entropy loss of the mapping of the received signals in high, normal, and low learning rate cases.

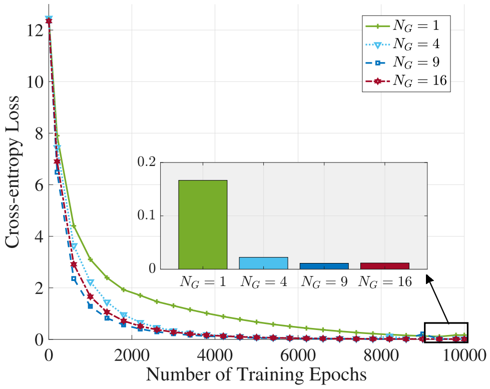

Fig. 11. Cross-entropy loss versus the number of training epochs for different sizes of the metasurface.

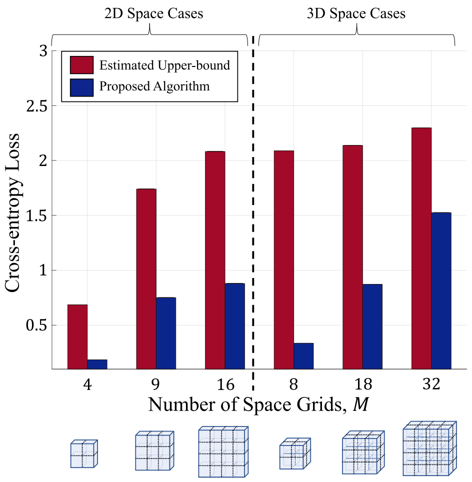

Fig. 12. Estimated upper-bound and the results of the proposed algorithm for the cross-entropy loss versus different numbers of space grids in 2D and 3D scenarios. The drawings at the bottom indicate the arrangement of the space grids.

Figure 10 shows the training results versus the number of training epochs for the PRPG algorithm are given and compared for different learning rates are compared. The initial learning rates in each case are set to be $\alpha_0 = 10^{-1}$, 10^{-3}, 10^{-5}, which then decrease inversely as the number of training epochs increases. It can be observed that large values of the learning rates prevent the algorithm to converge, while low values of the learning rates result in a slow decrease of the cross entropy loss. The setup $\alpha_0 = 10^{-3}$ outperforms the others, which verifies our learning rate selection.

In Fig. 11, it can be observed that as the size of the metasurface, i.e., N_G, increases, the cross-entropy loss after training decreases. This is because the received energy can be improved with more reconfigurable elements to reflect the transmitted signals, as indicated by (4). Besides, more reconfigurable elements create a larger design space and higher controllability of the beamforming, and thus beamformer patterns which are more effective for sensing can be adopted. Therefore, objects at different space grids can be sensed with a higher precision. However, the cross-entropy cannot be reduced infinitely. When N_G is sufficiently large, the cross-entropy remains stable. As shown in Fig. 11, the cross-entropy loss results for $N_G = 9$ and $N_G = 16$ are almost the same. Besides, comparing the curves for $N_G = 9$ and $N_G = 16$ within the first 2000 training epochs, we can observe that increasing the number of reconfigurable elements when $N_G \geq 9$ has a negative impact on the training speed and convergence rate. This is because increasing the number of reconfigurable elements leads to a higher complexity of finding the optimal policy for the metasurface to determine its control matrix, since the policy network of the metasurface needs to handle a higher-dimensional state space.

In Fig. 12, we compare the theoretical upper-bound derived in (46) and the proposed PRPG algorithm for different values of M in 2D and 3D scenarios. It can be observed that, in both 2D and 3D scenarios, the probability of sensing error increases with M. Also, the cross-entropy loss in 3D scenarios is higher than that for 2D scenarios. This is because the space grids in the 3D scenarios are more closely spaced to each other, which make them hard to be distinguished. Finally, it can be observed that, as M increases, the cross-entropy loss of the proposed algorithm increases more quickly in 3D scenarios compared to that in 2D scenarios. This which verifies that 3D sensing is more difficult than 2D sensing.

In Fig. 13, we show the comparison between the proposed metasurface-assisted scenario and the benchmark, which is the MIMO RF sensing scenarios with no metasurface. Both the metasurface-assisted scenario and the MIMO scenarios adopted a similar layout described in Section VI-A, and the result cross-entropy loss is obtained by Algorithm 1. Nevertheless, in the MIMO sensing scenarios, a static reflection surface takes the place of the metasurface, which cannot change the beamformer pattern for the reflection signals. When the size of the MIMO array in Fig. 13 is $n \times n$ ($n = 1, 2, \ldots, 5$), it indicates that n Tx antennas and n Rx antennas are adopted in the scenario. Specifically, the Tx/Rx antennas are arranged along the y-axis with a space interval of 0.1 m. The Tx antennas transmit continuous signals with phase interval $2\pi/n$, and the n received signals of the Rx antennas with suppressed LoS signals are used as the measurement vector. Comparing the MIMO benchmarks and the proposed method,

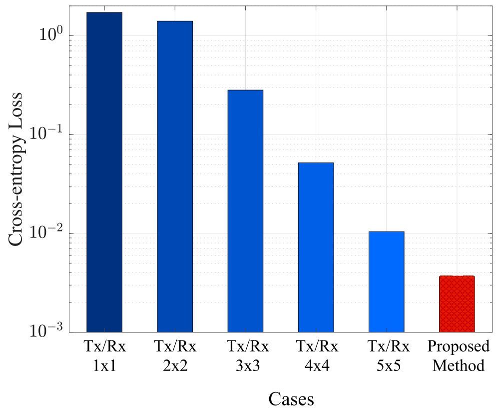

Fig. 13. Comparison between the metasurface-assisted scenario and the MIMO scenarios with different numbers of Tx/Rx antennas. The bars illustrate the results for different MIMO scenarios, and the dash lines depict the results of the metasurface-assisted scenario with different numbers of frames.

we can observe that the proposed method outperforms the $(1 \times 1) \sim (5 \times 5)$ MIMO sensing scenarios in terms of the resulting cross-entropy loss. This shows the significance of using metasurface to assisted RF sensing.

VII. CONCLUSION

In this paper, we have considered a metasurface-assisted RF sensing scenario, where the existence and locations of objects within a 3D target space can be sensed. To facilitate the beamformer pattern design, we have proposed a frame-based RF sensing protocol. Based on the proposed protocol, we have formulated an optimization problem to design the beamformer pattern and the mapping of the received signals. To solve the optimization problem, we have formulated an MDP framework and have proposed a deep reinforcement learning algorithm named PRPG to solve it. Also, we have analyzed the computational complexity and the convergence of the proposed algorithm and have computed a theoretical upper-bound for the cross-entropy loss given the beamformer patterns of the metasurface. Simulation results have verified that the proposed algorithm outperforms other benchmark algorithms in terms of training speed and cross-entropy loss. Besides, they have also illustrated the influence of the size of the metasurface and the target space on the cross-entropy loss, which provides insights on the implementation of practical metasurface-assisted RF sensing systems.

APPENDIX I
PROOF OF THEOREM 1

The complexities of embedding K and N as one-hot vectors is $\mathcal{O}(K)$ and $\mathcal{O}(N)$, respectively. Based on [44], the complexity of multiplex the K beamformer patterns that are $N \cdot N_S$-dimensional by $\boldsymbol{A} \in \mathbb{C}^{NN_S \times M}$ is $\mathcal{O}(K \cdot N \cdot N_S \cdot M)$. For a fully connected neural network with a fixed number of hidden layers and neurons, the computational complexity of the back-propagation algorithm is proportional to the product of the input size and the output size [45]. Therefore, the computational complexities of the symmetric MLP group and the Q-value MLP are $\mathcal{O}(K \cdot M)$ and $\mathcal{O}(K \cdot N_S + N \cdot N_S)$, respectively. As the connecting operation has complexity $\mathcal{O}(1)$ and finding the maximum Q-value is $\mathcal{O}(N_S)$, the total computational complexity has complexity $\mathcal{O}(KNN_SM)$ and is therefore dominated by the matrix multiplication. ■

APPENDIX II
PROOF OF LEMMA 1

We consider the worst case scenario for the computation, i.e., the former states in all the samples are terminal states. In this case, the rewards are calculated from (25). The term inside the second summation consists of two part, i.e., the cross-entropy calculation which has computational complexity $\mathcal{O}(M)$, and the calculation of $\hat{\boldsymbol{p}}$ by using the sensing network. The computational complexity of calculating $\boldsymbol{CA}$ is $\mathcal{O}(KNN_SM)$.

Based on [46], calculating the pseudo-inverse matrix $\boldsymbol{\Gamma}^{+}$, where $\boldsymbol{\Gamma}$ is a $K \times M$ matrix, has complexity $\mathcal{O}(K^2M)$. Similar to the analysis of the computational complexity of MLPs in the proof of Theorem 1, the computational complexity of the MLP is $\mathcal{O}(M^2)$. Therefore, the computational complexity of calculating $\hat{\boldsymbol{p}}$ is $\mathcal{O}(KNN_SM + K^2M + M^2)$, which proves Lemma 1. ■

APPENDIX III
PROOF OF LEMMA 2

For a fully connected neural network with a fixed number of hidden layers and neurons, the computational complexity of the back-propagation algorithm is proportional to the product of the input size and the output size [45]. Therefore, the computational complexity of using the back-propagation algorithm for updating the parameter vector of the sensing network is $\mathcal{O}(M^2)$.

The policy network can be considered as two connected MLPs: the first one takes the one-hot embedding vectors of k and n as the input, and the second one takes the K measuring vectors with $2M$ dimensions as the input. Moreover, as a symmetric MLP group is considered, the actual size of the input vector for the second MLP is $2M$ instead of $2KM$. Therefore, the computational complexity of training the first and second MLP of the policy network are $\mathcal{O}(N_S \cdot (K + N))$ and $\mathcal{O}(N_SM)$, respectively, and the total computational complexity is thus $\mathcal{O}(N_S \cdot (K + N + M))$.

Furthermore, if a single large MLP with layer sizes $(2KM, 64, 32K)$ is used to substitute the symmetric MLP group, the computational complexities of training the second MLP is $\mathcal{O}(KN_SM)$, and the total computational complexity of training the policy network is $\mathcal{O}(KMN_S + NN_S)$. Therefore, Lemma 2 is proved. ■

APPENDIX IV
PROOF OF THEOREM 2

Based on (26) and (27), the complexity of calculating the loss functions are determined by the computation of the reward and action probabilities. From Theorem 1 and Lemma 2, it follows that the complexity of calculating the reward is of higher order than that of calculating the action probabilities. Therefore, the computational complexity of the training phase is dominated by the calculation of the N_b rewards, which is $\mathcal{O}\big(KNN_SM + K^2M + M^2\big)$. Theorem 2 is thus proved. ■

REFERENCES

[1] S. Kianoush, S. Savazzi, F. Vicentini, V. Rampa, and M. Giussani, “Device-free RF human body fall detection and localization in industrial workplaces,” *IEEE Internet Things J.*, vol. 4, no. 2, pp. 351–362, Apr. 2017.

[2] P. W. Q. Lee, W. K. G. Seah, H.-P. Tan, and Z. Yao, “Wireless sensing without sensors—An experimental approach,” in *Proc. IEEE 20th Int. Symp. Pers., Indoor Mobile Radio Commun.*, Tokyo, Japan, Sep. 2009, pp. 62–66.

[3] T. He *et al.*, “VigilNet: An integrated sensor network system for energy-efficient surveillance,” *ACM Trans. Sensor Netw.*, vol. 2, no. 1, pp. 1–38, Feb. 2006.

[4] K. Ota, M. Dong, J. Gui, and A. Liu, “QUOIN: Incentive mechanisms for crowd sensing networks,” *IEEE Netw.*, vol. 32, no. 2, pp. 114–119, Mar. 2018.

[5] M. Schmitter-Edgecombe and D. J. Cook, “Assessing the quality of activities in a smart environment,” *Methods Inf. Med.*, vol. 48, no. 05, pp. 480–485, 2009.

[6] M. G. Amin, Y. D. Zhang, F. Ahmad, and K. C. D. Ho, “Radar signal processing for elderly fall detection: The future for in-home monitoring,” *IEEE Signal Process. Mag.*, vol. 33, no. 2, pp. 71–80, Mar. 2016.

[7] D. Zhang, J. Wang, J. Jang, J. Zhang, and S. Kumar, “On the feasibility of Wi-Fi based material sensing,” in *Proc. MobiCom*, Los Cabos, Mexico, Oct. 2019, pp. 1–16.

[8] W. Jiang *et al.*, “Towards environment independent device free human activity recognition,” in *Proc. MobiCom*, New Delhi, India, Oct. 2018, pp. 289–304.

[9] C.-Y. Hsu, R. Hristov, G.-H. Lee, M. Zhao, and D. Katabi, “Enabling identification and behavioral sensing in homes using radio reflections,” in *Proc. CHI*, Glasgow, U.K., May 2019, pp. 1–13.

[10] F. Adib, C.-Y. Hsu, H. Mao, D. Katabi, and F. Durand, “Capturing the human figure through a wall,” *ACM Trans. Graph.*, vol. 34, no. 6, p. 219, Oct. 2015.

[11] M. Zhao *et al.*, “RF-based 3D skeletons,” in *Proc. SIGCOMM*, New York, NY, USA, Aug. 2018, pp. 267–281.

[12] A. Pedross-Engel *et al.*, “Orthogonal coded active illumination for millimeter wave, massive-MIMO computational imaging with metasurface antennas,” *IEEE Trans. Comput. Imag.*, vol. 4, no. 2, pp. 184–193, Jun. 2018.

[13] N. Honma, D. Sasakawa, N. Shiraki, T. Nakayama, and S. Iizuka, “Human monitoring using MIMO radar,” in *Proc. iWEM*, Nagoya, Japan, Aug. 2018, pp. 1–2.

[14] Z. Li *et al.*, “Programmable radio environments with large arrays of inexpensive antennas,” *GetMobile, Mobile Comput. Commun.*, vol. 23, no. 3, pp. 23–27, Jan. 2020.

[15] M. Di Renzo *et al.*, “Smart radio environments empowered by AI reconfigurable meta-surfaces: An idea whose time has come,” 2019, *arXiv:1903.08925*. [Online]. Available: http://arxiv.org/abs/1903.08925 and https://jwcn-eurasipjournals.springeropen.com/articles/10.1186/s13638-019-1438-9

[16] H. Gacanin and M. Di Renzo, “Wireless 2.0: Towards an intelligent radio environment empowered by reconfigurable meta-surfaces and artificial intelligence,” 2020, *arXiv:2002.11040*. [Online]. Available: http://arxiv.org/abs/2002.11040 and https://ieeexplore.ieee.org/document/9205201

[17] E. Basar, M. Di Renzo, J. De Rosny, M. Debbah, M.-S. Alouini, and R. Zhang, “Wireless communications through reconfigurable intelligent surfaces,” *IEEE Access*, vol. 7, pp. 116753–116773, Aug. 2019.

[18] S. Zhang, H. Zhang, B. Di, Y. Tan, Z. Han, and L. Song, “Beyond intelligent reflecting surfaces: Reflective-transmissive metasurface aided communications for full-dimensional coverage extension,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13905–13909, Nov. 2020.

[19] M. A. ElMossallamy, H. Zhang, L. Song, K. G. Seddik, Z. Han, and G. Y. Li, “Reconfigurable intelligent surfaces for wireless communications: Principles, challenges, and opportunities,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 3, pp. 990–1002, Sep. 2020.

[20] H. Hashida, Y. Kawamoto, and N. Kato, “Intelligent reflecting surface placement optimization in air-ground communication networks toward 6G,” *IEEE Wireless Commun.*, vol. 27, no. 6, pp. 146–151, Dec. 2020.

[21] L. Li *et al.*, “Machine-learning reprogrammable metasurface imager,” *Nature Commun.*, vol. 10, no. 1, pp. 1–8, Dec. 2019.

[22] H. Zhang, H. Zhang, B. Di, K. Bian, Z. Han, and L. Song, “Towards ubiquitous positioning by leveraging reconfigurable intelligent surface,” *IEEE Commun. Lett.*, vol. 25, no. 1, pp. 284–288, Jan. 2021.

[23] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[24] Y. Liu *et al.*, “Reconfigurable intelligent surfaces: Principles and opportunities,” 2020, *arXiv:2007.03435*. [Online]. Available: http://arxiv.org/abs/2007.03435

[25] L. Dai *et al.*, “Reconfigurable intelligent surface-based wireless communications: Antenna design, prototyping, and experimental results,” *IEEE Access*, vol. 8, pp. 45913–45923, Mar. 2020.

[26] 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. Sel. Areas Commun.*, vol. 38, no. 8, pp. 1809–1822, Aug. 2020.

[27] J. Hu *et al.*, “Reconfigurable intelligent surface based RF sensing: Design, optimization, and implementation,” *IEEE J. Sel. Areas Commun.*, vol. 38, no. 11, pp. 2700–2716, Nov. 2020.

[28] A. Goldsmith, *Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[29] W. Tang *et al.*, “Wireless communications with reconfigurable intelligent surface: Path loss modeling and experimental measurement,” *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 421–439, Jan. 2021.

[30] H. Zhang, B. Di, L. Song, and Z. Han, “Reconfigurable intelligent surfaces assisted communications with limited phase shifts: How many phase shifts are enough?” *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4498–4502, Apr. 2020.

[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[32] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[33] R. N. McDonough and A. D. Whalen, *Detection of Signals in Noise*. San Diego, CA, USA: Academic, 2004.

[34] J. C. Bezdek and R. J. Hathaway, “Convergence of alternating optimization,” *Neural, Parallel Sci. Comput.*, vol. 11, no. 4, pp. 351–368, Dec. 2003.

[35] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[36] H. Li, K. Ota, and M. Dong, “Deep reinforcement scheduling for mobile crowdsensing in fog computing,” *ACM Trans. Internet Technol.*, vol. 19, no. 2, pp. 1–18, Apr. 2019.

[37] A. Zappone, M. Di Renzo, M. Debbah, T. T. Lam, and X. Qian, “Model-aided wireless artificial intelligence: Embedding expert knowledge in deep neural networks for wireless system optimization,” *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 60–69, Sep. 2019.

[38] A. Zappone, M. Di Renzo, and M. Debbah, “Wireless networks design in the era of deep learning: Model-based, AI-based, or both?” *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Oct. 2019.

[39] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo Method*. Hoboken, NJ, USA: Wiley, 2008.

[40] J. F. Bailyn, “Generalized inversion,” *Natural Lang. Linguistic Theory*, vol. 22, no. 1, pp. 1–50, Feb. 2004.

[41] F. J. Pineda, “Generalization of back-propagation to recurrent neural networks,” *Phys. Rev. Lett.*, vol. 59, no. 19, p. 2229, Jun. 1987.

[42] Y. Xu and W. Yin, “Block stochastic gradient iteration for convex and nonconvex optimization,” *SIAM J. Optim.*, vol. 25, no. 3, pp. 1686–1716, Jan. 2015.

[43] F. Hirtenfelder, “Effective antenna simulations using CST MICROWAVE STUDIO,” in *Proc. INICA*, Munich, Germany, Mar. 2007, p. 239.

[44] S. S. Skiena, *Sorting and Searching*. London, U.K.: Springer, 2012.

[45] M. Sipper, “A serial complexity measure of neural networks,” in *Proc. IEEE ICNN*, San Francisco, CA, USA, Mar. 1993, pp. 962–966.

[46] M. D. Petković and P. S. Stanimirović, “Generalized matrix inversion is not harder than matrix multiplication,” *J. Comput. Appl. Math.*, vol. 230, no. 1, pp. 270–282, Aug. 2009.

Jingzhi Hu (Graduate Student Member, IEEE) received the B.S. degree in electronic engineering from Peking University, China, in 2017, where he is currently pursuing the Ph.D. degree with the School of Electrical Engineering and Computer Science. His current research interests include reinforcement learning, compressive sensing, and wireless network virtualization.

Hongliang Zhang (Member, IEEE) received the B.S. and Ph.D. degrees from the School of Electrical Engineering and Computer Science, Peking University, in 2014 and 2019, respectively. From July 2019 to July 2020, he was a Postdoctoral Fellow with the Electrical and Computer Engineering Department, University of Houston, TX, USA. He is currently a Post-Doctoral Associate with the Department of Electrical Engineering, Princeton University, NJ, USA. His current research interests include cooperative communications, the Internet-of-Things networks, hypergraph theory, and optimization theory. He served as a TPC member for many IEEE conferences, such as Globecom, ICC, and WCNC. He received the Best Doctoral Thesis Award from the Chinese Institute of Electronics in 2019. He is currently an Editor of *IET Communications*. He also serves as a Guest Editor for IEEE INTERNET OF THINGS Special Issue on Internet of UAVs over Cellular Networks.

Kaigui Bian (Senior Member, IEEE) received the B.S. degree in computer science from Peking University, Beijing, China, in 2005, and the Ph.D. degree in computer engineering from Virginia Tech in 2011. He is currently an Associate Professor with the Department of Computer Science and Technology, School of EECS, Peking University. His research interests include wireless networking and edge computing. He received best paper awards of five international conferences. He was a recipient of the IEEE Communication Society Asia–Pacific Board (APB) Outstanding Young Researcher Award in 2018. He also serves as an Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY and IEEE ACCESS. He has been an IEEE Communication Society Distinguished Lecturer since 2020.

Marco Di Renzo (Fellow, IEEE) received the Laurea *(cum laude)* and Ph.D. degrees in electrical engineering from the University of L'Aquila, Italy, in 2003 and 2007, respectively, and the Habilitation à Diriger des Recherches (Doctor of Science) degree from University Paris-Sud (now Paris-Saclay University), France, in 2013. Since 2010, he has been with the French National Center for Scientific Research (CNRS), where he is a CNRS Research Director (a CNRS Professor) with the Laboratory of Signals and Systems (L2S), Paris-Saclay University – CNRS and CentraleSupelec, Paris, France. In Paris-Saclay University, he serves as the Coordinator of the communications and networks research area of the Laboratory of Excellence DigiCosme, and a member of the Admission and Evaluation Committee of the Ph.D. School on Information and Communication Technologies. He is a Fellow of the U.K. Institution of Engineering and Technology (IET). He has received several research distinctions, including the SEE-IEEE Alain Glavieux Award, the IEEE Jack Neubauer Memorial Best Systems Paper Award, the Royal Academy of Engineering Distinguished Visiting Fellowship, the Nokia Foundation Visiting Professorship, and the 2021 *EURASIP Journal on Wireless Communications and Networking* Best Paper Award. He is the Editor-in-Chief of IEEE COMMUNICATIONS LETTERS and a Distinguished Speaker of the IEEE Vehicular Technology Society. From 2017 to 2020, he was a Distinguished Lecturer of the IEEE Vehicular Technology Society and the IEEE Communications Society.

Zhu Han (Fellow, IEEE) received the B.S. degree in electronic engineering from Tsinghua University, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively.

From 2000 to 2002, he was a Research and Development Engineer of JDSU, Germantown, MD, USA. From 2003 to 2006, he was a Research Associate with the University of Maryland. From 2006 to 2008, he was an Assistant Professor with Boise State University, ID, USA. He is currently the John and Rebecca Moores Professor with the Electrical and Computer Engineering Department and the Computer Science Department, University of Houston, TX, USA. He is also a Chair Professor with National Chiao Tung University, China. He is also a 1% Highly Cited Researcher since 2017, according to Web of Science. His research interests include wireless resource allocation and management, wireless communications and networking, game theory, big data analysis, security, and smart grid. He received the NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the *EURASIP Journal on Advances in Signal Processing* in 2015, the IEEE Leonard G. Abraham Prize in communications systems (Best Paper Award in IEEE JSAC) in 2016, the 2021 IEEE Kiyo Tomiyasu Award for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: For contributions to game theory and distributed management of autonomous communication networks, and several best paper awards in IEEE conferences. He is also an IEEE Communications Society Distinguished Lecturer from 2015 to 2018.

Lingyang Song (Fellow, IEEE) received the Ph.D. degree from the University of York, U.K., in 2007. He worked as a Post-Doctoral Research Fellow with the University of Oslo, Norway, and Harvard University, until rejoining Philips Research, U.K., in March 2008. In May 2009, he joined the School of Electronics Engineering and Computer Science, Peking University, China, as a Full Professor. He published extensively, wrote six text books, and a Co-Inventor of a number of patents (standard contributions). His main research interests include cooperative and cognitive communications, physical layer security, and wireless ad hoc/sensor networks. He received nine paper awards in IEEE journal and conferences, including IEEE JSAC 2016, IEEE WCNC 2012, ICC 2014, Globecom 2014, and ICC 2015, the K. M. Stott Prize for Excellent Research from the University of York, and the 2012 IEEE Asia Pacific (AP) Young Researcher Award. He served as the TPC Co-Chair for the International Conference on Ubiquitous and Future Networks (ICUFN2011/2012). He served as the Symposium Co-Chair for the International Wireless Communications and Mobile Computing Conference (IWCMC 2009/2010), the IEEE International Conference on Communication Technology (ICCT2011), and the IEEE International Conference on Communications (ICC 2014, 2015). He is currently on the Editorial Board of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and *Journal of Network and Computer Applications*. He is also a IEEE ComSoc Distinguished Lecturer since 2015.