Offline Fictitious Self-Play for Competitive Games

Jingxiao Chen[1], **Weiji Xie**[1], **Weinan Zhang**[1*], **Yong Yu**[1], **Ying Wen**[1*]

[1]Shanghai Jiao Tong University
timemachine@sjtu.edu.cn, wnzhang@sjtu.edu.cn, ying.wen@sjtu.edu.cn

Abstract

Offline Reinforcement Learning (RL) enables policy improvement from fixed datasets without online interactions, making it highly suitable for real-world applications lacking efficient simulators. Despite its success in the single-agent setting, offline multi-agent RL remains a challenge, especially in competitive games. Firstly, unaware of the game structure, it is impossible to interact with the opponents and conduct a major learning paradigm, self-play, for competitive games. Secondly, real-world datasets cannot cover all the state and action space in the game, resulting in barriers to identifying Nash equilibrium (NE). To address these issues, this paper introduces OFF-FSP, the first practical model-free offline RL algorithm for competitive games. We start by simulating interactions with various opponents by adjusting the weights of the fixed dataset with importance sampling. This technique allows us to learn the best responses to different opponents and employ the Offline Self-Play learning framework. To overcome the challenge of partial coverage, we combine the single-agent offline RL method with Fictitious Self-Play (FSP) to approximate NE by constraining the approximate best responses away from out-of-distribution actions. Experiments on matrix games, extensive-form poker, and board games demonstrate that OFF-FSP achieves significantly lower exploitability than state-of-the-art baselines. Finally, we validate OFF-FSP on a real-world human-robot competitive task, demonstrating its potential for solving complex, hard-to-simulate real-world problems.

1 Introduction

Multi-agent reinforcement learning (MARL) provides a powerful learning framework to tackle the problems in multi-agent systems and has been applied to a wide range of domains, such as Go (Silver et al. 2017), strategy games (Vinyals et al. 2019), robotics (Yu et al. 2023), unmanned aerial vehicle (Yun et al. 2022), and network routings (Ye, Zhang, and Yang 2015). MARL typically relies on extensive interaction and exploration with accurate and efficient environments, hindering its application in the real world. In many real-world multi-agent problems, such as football (Kurach et al. 2020), negotiation (Yang, Chen, and Narasimhan 2020), and crowdsourcing (Gerstgrasser, Trivedi, and Parkes 2021), the absence of reliable simulators makes it expensive and inefficient to interact with the environment and collect new data for training MARL agents. Moreover, in special problems, such as wildlife protection (Fang et al. 2016), learning MARL agents online by trial and error is unsafe, which could lead to danger for patrols or wildlife. In these cases, learning from existing data can be extremely valuable as it does not require additional sampling. Offline MARL offers an excellent alternative to solve these issues by improving policies from previously collected datasets without further interactions (Tseng et al. 2022; Yang et al. 2021).

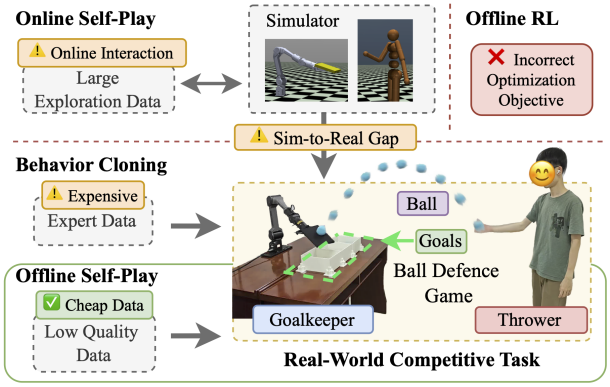


Figure 1: Comparison of offline self-play with other learning paradigms for real-world competitive tasks. Online self-play requires a simulator and suffers from the sim-to-real gap. Behavior cloning needs costly expert data. Naive offline RL optimizes with incorrect objectives. In contrast, OFF-SP learns from inexpensive, low-quality data. See Section 5.2 for experiments on the illustrated real-world game.

In competitive multi-agent games, often framed as zero-sum games, the objective is to find a Nash equilibrium (NE) (Kreps 2018) or maximizing the ELO ratio (Silver et al. 2017; Ye et al. 2020), which is divergent from the goal of maximizing cumulative rewards in single-agent situations. Exemplified by studies of Texas hold'em poker (Brown and Sandholm 2018, 2019), the pursuit of a Nash equilibrium enables agents to learn robust policies against various opponents, enhancing their perfor-

*Correspondence to: Weinan Zhang and Ying Wen.

mance in competitive environments. To achieve the goals, online approaches predominantly rely on the self-play paradigm (Zhang et al. 2024), wherein policies are continuously refined to maximize returns against evolving adversary policies. However, in the offline scenario, the challenge arises from the absence of online interactions with evolving opponents, complicating the development of self-play paradigms. Recent benchmarks of offline competitive games, such as AlphaStar Unplugged (Mathieu et al. 2023) and Hokoff (Qu et al. 2023; Wei et al. 2022), have directly applied single-agent offline RL algorithms with no self-play. However, maximizing the cumulative rewards against static opponents results in an overfitting policy and vulnerable exploitation by more dynamic opponents. Moreover, when datasets are collected by poor policies, such as random policies, it is pointless to learn to defeat these weak opponents, because real-life opponents typically exhibit rational and high-performance behaviors.

The reliance on high-quality or high-coverage datasets, which are commonly expensive and suboptimal in the real world, is another challenge for offline learning. The existing data-driven method, supervised learning, also called behavior cloning, ignores the objective of online learning and only imitates the sampling policy of datasets. Consequently, this method performs poorly when dealing with non-expert datasets. Many recent works are learning to improve policies toward the NE, but they demand high state-action coverage within the dataset. Li et al. (2022) proposes a model-based offline paradigm for equilibrium finding, but it requires a strong assumption that datasets fully cover the state-action space of the original games. Cui and Du (2022b); Zhong et al. (2022); Cui and Du (2022a) contribute to theoretical insights for a weaker assumption on datasets but only propose theoretically feasible algorithms. Existing work lacks a practical method applicable to non-expert and partially covered real-world datasets. Figure 1 compares our paradigm with existing works, highlighting the advantage of OFF-SP on solving real-world competitive problems. Detailed comparisons with related works are provided in the appendix.

In this paper, we propose an offline learning framework, called **Offline Self-Play (OFF-SP)**, and an offline learning algorithm, **Offline Fictitious Self-Play (OFF-FSP)**, for equilibrium finding in zero-sum extensive-form games, bridging the gap between single-agent offline RL and competitive games. To our knowledge, OFF-FSP is the first model-free offline algorithm for practical zero-sum games that offers the flexibility to combine with various Offline RL agents and improve policies on non-expert real-world datasets. We first propose a technique to approximate interaction with different opponents by re-weighting the datasets with importance sampling. This allows us to learn the approximate best responses against arbitrary opponents with offline RL and derive a self-play paradigm under the offline setting. Also, for partially covered and non-expert datasets, we use a surrogate loss, NashConv, to measure the distance to NE, rather than finding the exact NE. OFF-FSP combines single-agent offline reinforcement learning methods with fictitious self-play to learn approximate best responses iteratively and minimize the NashConv. We validate our approach across matrix-form and extensive-form zero-sum games, demonstrating consistently low exploitability and superior performance over existing offline RL baselines under partially covered datasets. Notably, we further apply our method to a real-world human-robot competitive task, showcasing its potential in addressing hard-to-simulate decision-making problems.

2 Preliminaries

2.1 Extensive-form Game

Extensive-form games are a model of sequential interaction involving n agents. Each player's goal is to maximize his payoff in the game. At each step t of extensive-form game, only one player observes his respective **information states** $s_t^i \in \mathcal{S}^i$ and suggests his **action** $a_t^i \in \mathcal{A}^i(s_t^i)$. $\mathcal{S}^i$ is the set of information states of player i, and $\mathcal{A}^i(s)$ is the set of available action at state s. The **player function** $\mathcal{P} : \mathcal{S} \to \mathcal{N}$, with $\mathcal{N} = \{1, \dots, n\}$ denotes the set of players, determines the player to act.

In extensive-form games, each player plays following a **policy** $\pi^i : \mathcal{S}^i \to \Delta(\mathcal{A}^i)$ that maps information states to distributions of actions. The **realization-plan** (Von Stengel 1996), $x(s_t^i) = \Pi_{j=1}^{t-1} \pi^i(a_j^i | s_j^i)$, describes the probability of reaching the information state s_t^i following player i's policy, π^i. The **strategy profile** $\pi = (\pi^1, \dots, \pi^n)$, is a joint of all player's policy. π^{-i} denotes the strategy profile of all players excepts i. The **payoff** of player i is denoted by $R^i(\pi^i, \pi^{-i}) \in \mathbb{R}$, and $\sum_i R^i = 0$ for zero-sum games. Given a fixed π^{-i}, **best response** $\mathrm{BR}(\pi^{-i})$ is the policy with the highest payoff. A **Nash equilibrium** (NE) is a strategy profile that any policy π^i in this profile is a BR to the opponent's profile π^{-i}. The **ϵ-best response** (ϵ-BR) and **ϵ-Nash equlibirium** (ϵ-NE) are approximations to the above definition. ϵ-BR is suboptimal by no more than ϵ compared with BR. Similarly, ϵ-NE is a profile of ϵ-BR. **NASHCONV** (Timbers et al. 2022), also called exploitability in two-player zero-sum games, evaluates the distance from π to an NE, defined as $\sum_i R^i(\mathrm{BR}(\pi^{-i}), \pi^{-i}) - R^i(\pi^i, \pi^{-i})$.

2.2 Fictitious Self-Play

Fictitious Self-Play (FSP)(Heinrich, Lanctot, and Silver 2015) is a game-theoretic model that iteratively computes the best responses to opponents' average policy and updates their set of policies. We briefly describe FSP at appendix A. In extensive-form games, the average policies π_k^i are updated by the realization-equivalence theorem. At the k-th iteration of FSP, the average policy π_k^i of player i is

$$\begin{aligned} \forall s, a : \pi_k^i(a|s) &= (1-\lambda)\pi_{k-1}^i(a|s) + \lambda \beta_k^i(a|s), \\ \lambda &= \frac{\alpha_k x_{\beta_k^i}(s)}{(1-\alpha_k) x_{\pi_{k-1}^i}(s) + \alpha_k x_{\beta_k^i}(s)}, \end{aligned} \tag{1}$$

where $\beta_k^i \in \epsilon_k\text{-BR}(\pi_{k-1}^{-i})$ is a best-response to opponent π_{k-1}^{-i} and α_k is the mixing parameter. Policy π_k^i is equivalent to choosing either policy π_{k-1}^i or β_k^i before the beginning of each game, with probabilities $1-\alpha_k$ and α_k respectively. A standard choice is $\alpha_k = \frac{1}{k}$.

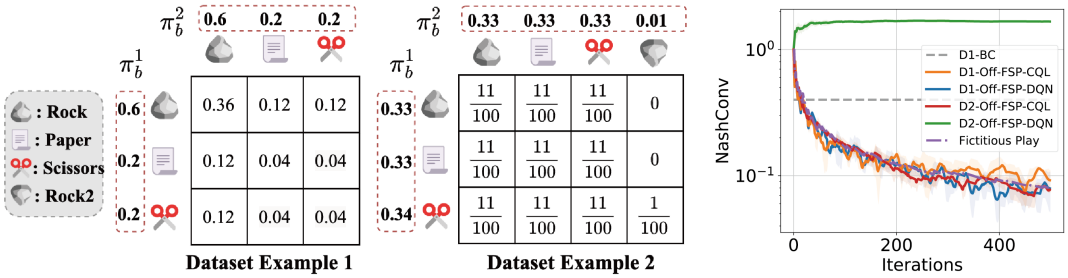


Figure 2: The Motivating RPS Example. **(Left):** Example Datasets of RPS. Numbers in the grids show the probability density of different samples. The red dashed boxes indicate the probability of different actions for corresponding behavioural policies. **(Right):** Results of RPS. The prefixes D1- and D2- refer to results on the first and second datasets, respectively.

2.3 Offline Reinforcement Learning

Reinforcement learning (RL) agents aim to maximize the expected cumulative discounted reward in a Markov decision process (MDP). MDP is denoted as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, r, \gamma, T)$. $\mathcal{S}, \mathcal{A}$ represent the state and action spaces. $\mathcal{T}(s_{t+1}|s_t, a_t), r(s_t, a_t)$ represent the dynamics and reward function. ρ_0 is the distribution of initial state $s_1 \sim \rho_0$. $\gamma \in [0, 1]$ is the discount factor. The expected cumulative discounted reward following policy π can be formalized as the action-value function $Q(s_t, a_t) = \mathbb{E}_\pi[\sum_{i=t}^{\infty} \gamma^{i-t} r(s_i, a_i)]$. Given the fixed opponent π_{-i} and an extensive form game, the game of player i can be defined as a MDP $\mathcal{M}(\pi_{-i})$ (Silver and Veness 2010; Greenwald et al. 2013). An ϵ-optimal policy of the MDP, $\mathcal{M}(\pi^{-i})$, is also the ϵ-BR to the policy π^{-i}.

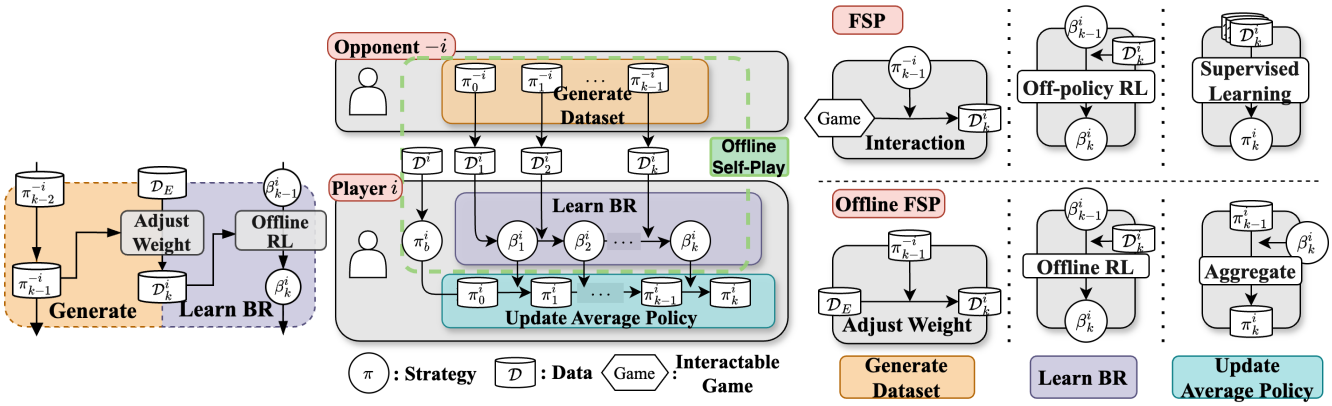
Offline RL algorithm breaks the assumption that the agent can interact with the environment. The offline RL algorithm learns to maximize the expected cumulative reward based on a fixed dataset. The dataset is sampled following a behavior policy $\pi_b(a|s)$ in MDP $\mathcal{M}$. $\hat{\pi}_b(a|s) := \frac{\sum_{s,a \in \mathcal{D}^i} \mathbb{1}[s=s, a=a]}{\sum_{s \in \mathcal{D}^i} \mathbb{1}[s=s]}$ denote the empirical behavior policy, at all state $s \in \mathcal{D}^i$. The Q-function may be erroneously overestimated at out-of-distribution (O. O. D.) actions, which is called extrapolation error (Fujimoto, Meger, and Precup 2019). Recent offline RL methods encourage the policy to learn on the support of training data or employ weighted behavior cloning to mitigate this error.

3 The Motivating Example: Rock-Paper-Scissors

We start by analysing the well-known game, Rock-Paper-Scissors (RPS), to illustrate the challenges of learning in offline datasets of competitive games. In this game, two players can choose one of three actions: Rock, Paper, or Scissors. The payoffs are defined as follows: Rock beats Scissors, Scissors beats Paper, and Paper beats Rock. The game has a unique Nash equilibrium (NE) where both players play each action with equal probability of $\frac{1}{3}$. In Section 2.2, we show two datasets of RPS. The first dataset is a fully covered dataset sampled from a non-expert policy, $P(\text{R,P,S}) = (0.6, 0.2, 0.2)$. The second dataset is a partially covered dataset sampled from a modified asymmetric RPS game, where the second player has a new action, Rock2, with the same payoff as Rock.

In the first non-expert dataset, neither Behavioral Cloning (BC) nor naive offline RL can find the NE, which highlights the importance of the self-play paradigm. BC mimics the distribution of the dataset by supervised learning. Its policy, $P_{\text{BC}}(\text{R,P,S}) = (0.6, 0.2, 0.2)$, is suboptimal and beaten by the policy of playing Paper only. As the solution of Mathieu et al. (2023); Qu et al. (2023), single-agent offline RL maximizes the payoff under the fixed opponent in the dataset, and its policy, $P_{RL}(\text{R,P,S}) = (0, 1, 0)$, is also easy to be exploited by another policies. Under the self-play paradigm, the policy learns to play against a dynamic opponent, which is more robust and can approximate the NE. Section 2.2 compares the performance of our method, OFF-FSP-CQL, with the baselines, BC, OFF-FSP-DQN and online Fictitious Play (FP) (Brown 1951).

Corresponding to the challenge of learning in partially covered datasets, the second dataset in Section 2.2 leaves the payoff of Rock2 not fully observed. In such datasets, finding the accurate NE of the original game is impossible , so the goal of learning is to minimize the exploitability and NashConv, i.e., the distance to NE. In the dataset, the only observed sample of Rock2 is $(\text{Scissors}, \text{Rock2})$. In player 2's perspective, the Rock2 action gets a payoff of 1 all the time, so choosing it with a probability of 1 is the best choice. Nevertheless, the policy has the highest exploitability in the original game, as player 1 possesses a best response by selecting scissors, resulting in a payoff of 1 for player 1. Offline RL algorithms, such as Conservative Q-Learning(CQL) (Kumar et al. 2020), mitigate this problem by punishing the agent for learning OOD actions, including unseen actions and undersampled actions. By combining self-play and offline RL, OFF-FSP ignores the samples with large uncertainty and learns robust policies with low exploitability. As shown in Section 2.2, OFF-FSP without offline RL (OFF-FSP-DQN) converges to high exploitability, while OFF-FSP with CQL can reduce the exploitability to a low level. We also show the learning curve of an online algorithm, FP, which shows the convergence speed of OFF-FSP is comparable to online algorithms. Further explanations of the results are given in the appendix.



(a) Offline Self-Play framework. (b) Pipeline of OFF-FSP. (c) Differences between FSP and OFF-FSP.

Figure 3: Illustration of OFF-SP, OFF-FSP and three essential steps. The green box in (b) is OFF-SP.

4 Offline Fictitious Self-Play

In this section, we introduce the Offline Self-Play (OFF-SP) learning framework and give an implementation as Offline Fictitious Self-Play (OFF-FSP) algorithm to minimize NashConv, the distance to NE, with a fixed dataset. OFF-SP learns policies iteratively, maximizing cumulative rewards against changing opponents without interactable environments. OFF-FSP adopts the fictitious self-play on OFF-SP, where the policy plays against the average of past opponents.

Initially, we describe the offline datasets and make assumptions to simplify the problem. Based on the original FSP, as described in Appendix A, we derive the offline fictitious self-play with modifications on three essential functions, *GenerateData*, *LearnBestResponse*, and *UpdateAveragePolicy*. In OFF-SP, *GenerateData* simulates play against different opponents with weighted datasets, and *LearnBestResponse* optimizes the policy with an existing single-agent offline RL algorithm. Offline RL algorithm, such as CQL, ensures the performance of BR and reduction of NashConv even on partially covered datasets. In Section 4.3, we introduce *UpdateAveragePolicy* by computing the average policy on samples and derive OFF-FSP.

4.1 Problem Formulation

The trajectory of extensive-form games is a sequence of information states, actions, and rewards. The trajectory is $\tau_E = \{s_1, a_1, r_1, \ldots, s_T, a_T, r_T\}$. In player i's perspective, the extensive-form game can be modeled as an MDP $\mathcal{M}$ given a fixed opponent π^{-i} (Silver and Veness 2010; Greenwald et al. 2013). In MDP $\mathcal{M}$, the dynamic function $\mathcal{T}(s^i_{t+1} \mid s^i_t, a^i_t)$ models the dynamics of opponents.

The goal of OFF-FSP is to iteratively learn the best-response policy π^i against changing opponents and minimize NashConv. To learn best-response with single-agent RL, we project the trajectory τ_E into player i's perspective with a projection function $\tau^i = \mathcal{F}^i(\tau_E)$. The projection $\mathcal{F}^i$ filters out the states corresponding to player i while relabeling the time indices.

$$\begin{aligned}\tau^i = \mathcal{F}^i(\tau_E) &= \{s_t, a_t, r_t \mid \forall s_t \in \tau_E, \mathcal{P}(s_t) = i\} \\ &= \{s^i_1, a^i_1, r^i_1, s^i_2, \ldots, s^i_{T'}, a^i_{T'}, r^i_{T'}\},\end{aligned}$$

where T' is the length of the player i's trajectory $\tau^i_{\mathcal{M}}$.

The subscripts of s_t and s^i_t are different and represent the time indices in the corresponding trajectories. For state $s^i_t \in \tau^i_{\mathcal{M}}$, we use function $I(s^i_t)$ to denote the subscript of state in τ_E, and $\tau^j_<(s^i_t)$ denotes opponent j's latest state at s^i_t.

$$\tau^j_<(s^i_t) = s_k, \text{ where } k = \max\{k' < I(s^i_t) \mid p_{k'} = j\}.$$

To offer a clear explanation of the stated components, we present an example trajectory in Figure 4. Based on the description of trajectory in both extensive-form games and single-agent perspective, the datasets consist of multiple corresponding trajectories. For extensive-form game, the dataset is $\mathcal{D}_E = \{\tau_E\}$. The dataset for player i is $\mathcal{D}^i = \{\tau^i_{\mathcal{M}} = \mathcal{F}^i(\tau_E) \mid \tau_E \in \mathcal{D}_E\}$. The i-th player's dataset, derived from dataset $\mathcal{D}_E$, is denoted as $\mathcal{D}^i = \mathcal{F}^i(\mathcal{D}_E)$.

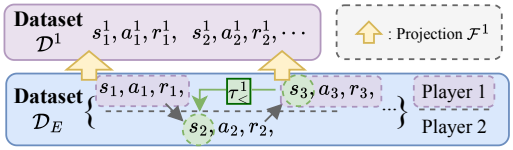


Figure 4: An illustration example of trajectory τ^1 and τ_E. Purple parts represent Player 1. The yellow arrows imply the projection relationship under $\mathcal{F}^1$. Green part indicates $\tau^1_<$.

To introduce the theoretical foundation of our method, we begin by defining an idealized dataset condition that facilitates analysis. These assumptions are not required for practical application and are used only to isolate and clarify the core mechanisms of our approach. In later sections, we show how the method generalizes to realistic datasets with partial coverage by integrating standard offline RL algorithms.

Definition 4.1 (Fully Covered Dataset)**.** $\mathcal{S}$ and $\mathcal{A}$ represent the joint sets of information states and actions for all players $\mathcal{N}$. A dataset $\mathcal{D}_E$ in extensive-form games is a **fully covered dataset** if $\forall s \in \mathcal{S}, a \in \mathcal{A}(s) \Rightarrow (s, a) \in \mathcal{D}_E$.

When a dataset is fully covered, the trained learning algorithm will not suffer from OOD problems.

Definition 4.2 (Real-Equivalence Dataset)**.** For a dataset of extensive-form game $\mathcal{D}_E$ sampled following a policy π_b, dataset $\mathcal{D}_E$ is an **real-equivalence dataset** if $Pr(\tau_E) = \frac{\sum_{\tau \in \mathcal{D}_E} \mathbb{1}[\tau = \tau_E]}{|\mathcal{D}_E|}, \forall \tau_E \in \mathcal{D}_E$, where $Pr(\tau_E)$ is the probability of sampling trajectory τ_E with π_b.

When a dataset is a real-equivalence dataset , sampling trajectories from the offline dataset $\mathcal{D}_E$ is equivalent to sampling from the extensive-form game with policy π_b online and the right-hand side of the equation is the probability density of τ_E in $\mathcal{D}_E$, denoted by $\mathcal{D}_E(\tau_E)$.

4.2 Offline Self-Play

Self-play iteratively interacts with the game to generate data and utilizes the data to learn the best responses. In k-th iteration, the opponent for player i is changed to π_{k-1}^{-i}, therefore, in order to learn the best response $\beta_k^i = \text{BR}(\pi_{k-1}^{-i})$, player i must interact with π_{k-1}^{-i} to generate new data $\mathcal{D}_k^i$. The process of learning BR, from the perspective of player i, is equivalent to interacting with a new MDP $\mathcal{M}(\pi_k^{-i})$ and maximizing returns with RL (Greenwald et al. 2013).

In offline settings, we are limited to using a fixed dataset sampled by behavioural policy π_b, in which RL methods can only obtain the best response $\text{BR}(\pi_b^{-i})$ for player i. In order to execute the fictitious self-play, we generate player i's datasets under different MDP $\mathcal{M}(\pi_k^{-i})$. With importance sampling (Nachum et al. 2019), we can emulate sampling from another dataset $\mathcal{D}_w^i$ with weighting $w(d_t) = \frac{\mathcal{D}_w^i(d_t)}{\mathcal{D}^i(d_t)}$ (Hong et al. 2023) on the original dataset $\mathcal{D}^i$, where $\mathcal{D}_w^i(d_t)$ and $\mathcal{D}^i(d_t)$ denote the probability density of tuple $d_t = (s_t^i, a_t^i, s_{t+1}^i)$. This formulation gives the following equivalence:

$$\mathbb{E}_{d_t \sim \mathcal{D}_w^i}[L(d_t; \theta)] \Leftrightarrow \mathbb{E}_{d_t \sim \mathcal{D}^i}[w(d_t) L(d_t; \theta)], \qquad (2)$$

where $L(d_t; \theta)$ is the loss function of an off-policy RL method, and θ is the parameter of the RL policy to be optimized. With the assumption that dataset $\mathcal{D}_E$ is both a fully covered dataset and a real-equivalence dataset, sampling from it with importance sampling is equivalent to sampling from the online game with importance sampling.

Theorem 4.3. *For player i, the weight of transferring the opponent from π_b^{-i} to π^{-i} is:*

$$w(d_t) = \frac{x_{\pi^{-i}}^{-i}(s_j)\pi^{-i}(a_j|s_j)}{x_{\pi_b^{-i}}^{-i}(s_j)\pi_b^{-i}(a_j|s_j)}, \; s_j = \tau_<^{-i}(s_{t+1}^i). \qquad (3)$$

The proof of Theorem 4.3 is provided in Appendix C. Given an offline dataset $\mathcal{D}_E$, the policy π_b can be approximated by an empirical behavioural policy $\tilde{\pi}_b$. We can estimate $\tilde{\pi}_b$ by counting or supervised learning policy.

In the learning process, the weight $w(d_t)$ assigned to a sample d_t may shift to zero or a large value. As a result, the RL loss suffers from a large variance (Munos et al. 2016), leading to unstable training. To address this issue, we employ $\frac{w(d_t)}{\sum_{d \in \mathcal{D}} w(d)}$ as the sampling probability rather than multiplying the loss function by $w(d_t)$ as shown in Equation 2. In this way, re-sampling from fully covered real-equivalence datasets is identical to sampling data in online games. With batches of data sampled from the dataset, we can apply an offline RL to learn BR.

Figures 3a and 3c illustrates these two functions and OFF-SP. In the function *GenerateData*, we first calculate $w(d)$ for all the samples d_t and get a re-weighted dataset $\mathcal{D}_k^i$ for each player i. In the function *LearnBestResponse*, offline RL algorithms repeatedly sample a batch of data and optimize the learned policy for M times. We use CQL (Kumar et al. 2020) as the default offline RL algorithm in our implementation, i.e., $\beta_k^i = \text{CQL}(\mathcal{D}_k^i)$. These two functions derive the OFF-SP. At each step t, OFF-SP first generates a dataset playing against the current opponents and updates policies towards the best response of the opponents.

4.3 Aggregate Average Policy in Samples

Similar to FSP, OFF-FSP play against the average policy π_{k+1} following Equation (1). Although an interactable average policy π_k is not necessary in the offline setting, we can just maintain the probability $\pi_k(s, a)$ in the datasets for the weighting technique. Traverse along the trajectory τ_i, OFF-FSP computes the probability $\pi_k^i(s, a)$ for player i following Equation (1). Figure 3c shows one step of the function *UpdateAveragePolicy*. To facilitate the calculation, we save the probability into the current dataset $\mathcal{D}_k^i$, changing the sample into $d' = (s, a, \pi_k^i(s, a))$. In short, we have:

$$\pi_k^i(s, a) \leftarrow \frac{k-1}{k} x_{\pi_{k-1}^i}^i(s)\pi_{k-1}^i(s, a) + \frac{1}{k} x_{\beta_k^i}^i(s)\beta_k^i(s, a).$$

In the evaluation phase, we aggregate all policies within a collection Π. Before the beginning of each game, one of the policies in Π is chosen with a specific probability, and the payoff of the aggregation is the expected payoff over all possible policies. To keep the storage efficient, we only keep policies at fixed interval steps for the real-world application in Section 5.2. Algorithm 2 in Appendix A and Figure 3b presents the pipeline of Offline Fictitious Self-play.

Memory and Computational Complexity. Compared with offline RL, OFF-FSP only assigns additional weights for samples in the dataset, so the memory complexity is still $O(|\mathcal{D}_E|)$ in training. The training time of OFF-FSP can be noted as $T = T_{reweight} + T_{learn}$, where $T_{reweight}$ is the time of re-weighting and T_{learn} is the time of offline RL learning. In empirical experiments of Section 5, the $T_{reweight} : T_{learn} \approx 1 : 1$.

5 Experiments

In this section, we design experiments to show the performance of OFF-FSP in offline competitive games. Three benchmark extensive-form games are selected for analysis: Leduc Poker, Large Kuhn Poker, and Oshi Zumo, which

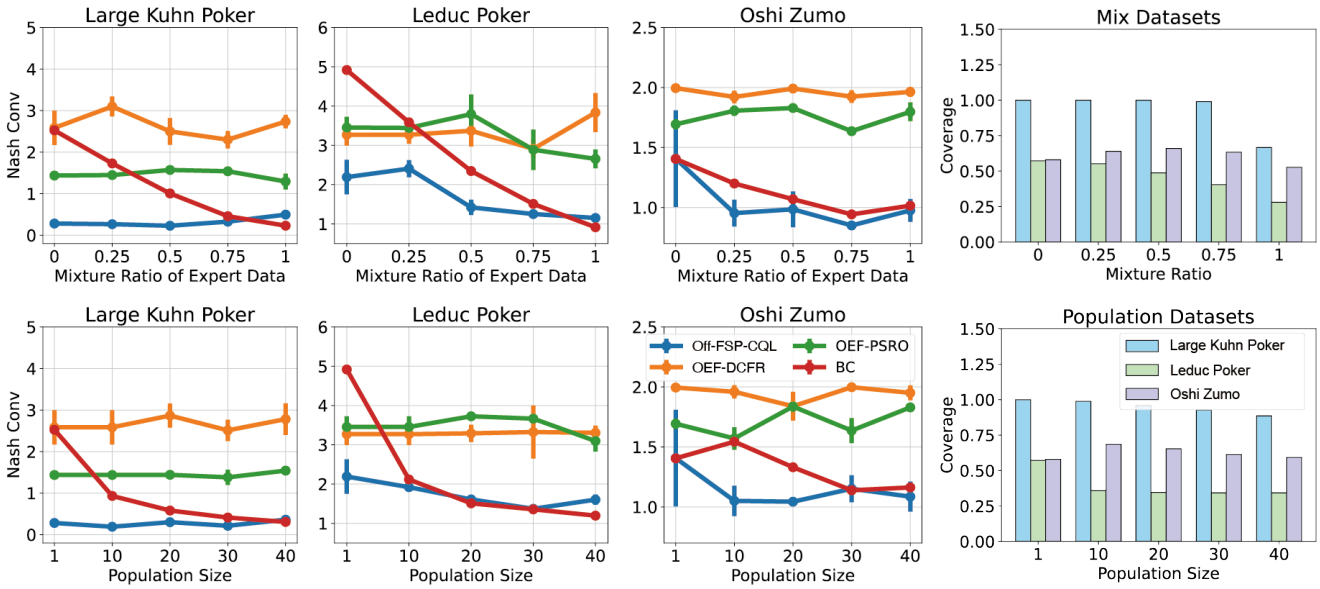


Figure 5: Results on Extensive-Form Games. **(Left Top)** NashConv on Mix Datasets; **(Left Bottom)** NashConv on Population Datasets; **(Right)** Coverage ratio of datasets.

include two stochastic poker games and one deterministic board game. These environments allow efficient sampling and easy NashConv computation, facilitating a comprehensive assessment of algorithm performance. We compare OFF-FSP with state-of-the-art baselines, including OEF (Li et al. 2022) and Behavior Cloning (BC). We further conduct ablations on offline RL components to demonstrate the flexibility of OFF-FSP. Finally, the practical applicability of OFF-FSP is highlighted through deployment in a real-world human-robot competitive task, showcasing its potential for solving complex, hard-to-simulate problems.

5.1 Extensive-form Games

To further evaluate OFF-FSP on complicated extensive-form games, we collected datasets of different quality on Leduc Poker, Large Kuhn Poker, and Oshi Zumo. Following the previous work (Li et al. 2022), the first type of datasets is called mix datasets, which are sampled from a mixture of expert and random policies. Evaluation on random, mixed, and expert datasets is also a common practice in single-agent offline RL (Kumar et al. 2020). We sample 5 different mix datasets for each game, the ratio of sampling from the expert are $0, 0.25, 0.5, 0.75, 1$. The expert policy is learned by online PSRO (Lanctot et al. 2017) with 40 iterations. The second type of dataset is called the population dataset, which is sampled from the population of online PSRO. 5 population datasets are uniformly sampled from all population policies of $1, 10, 20, 30, 40$ iterations of online PSRO. This type of dataset is designed to simulate the datasets sampled from a population of people with different levels of expertise and is similar to the setting in the real world. The population datasets with 1 iteration and the mix datasets with a ratio of 0 are random datasets. Every dataset comprises 10 000 trajectories, which is far less than the number in the online PSRO. We visualize the coverage of terminal states, i.e., leaf nodes, in Figure 5. Most of the datasets are partially covered.

Figure 5 shows the NashConv of OFF-FSP and baselines on three extensive-form games. The difficulty of learning from datasets is affected by two factors: the quality of sampling policies and the coverage of datasets. NashConvs of BC show the quality of sampling policies. In most cases, OFF-FSP shows the best performance, and the performance of OFF-FSP is also robust to both the quality of sampling policies and the coverage of datasets. Both OEF-DCFR and OEF-PSRO, two variants of OEF (Li et al. 2022), fail in most cases. OEF seeks an offline equilibrium using the model-based paradigm, but it is easy to be misled by O.O.D. actions. OEF mixed its policy with BC, evaluated it in online games multiple times, and used the minimum NashConv as a result. To make a fair comparison, we removed the mixing operation in OEF from our main experiment. Compared with BC, OFF-FSP outperforms in non-expert datasets and achieves comparable performance in expert datasets. In expert datasets, sample diversity is limited, and few samples can be used to improve the policy. OFF-FSP aims to improve the policy with non-expert datasets, which is more practical and easier to scale up in real-world problems.

Ablation Studies. In previous experiments, we showed the performance of OFF-FSP with CQL as the offline RL algorithm to learn the best response. However, the choice of an offline RL algorithm is also an important factor. Figure 6 shows learning curve of OFF-FSP with different RL algorithms, including CQL, BCQ (Fujimoto, Meger, and Precup 2019), CRR (Wang et al. 2020), and DQN on two datasets of Leduc Poker. Without the constraint of offline RL to avoid O.O.D. actions, OFF-FSP with DQN converges to a pol-

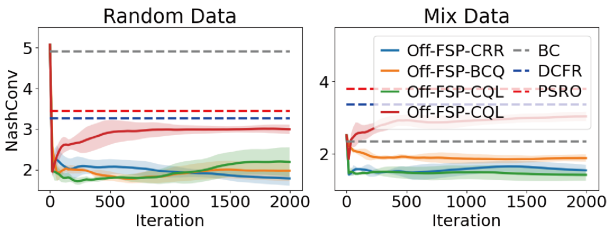


Figure 6: Results of ablation study on Leduc Poker.

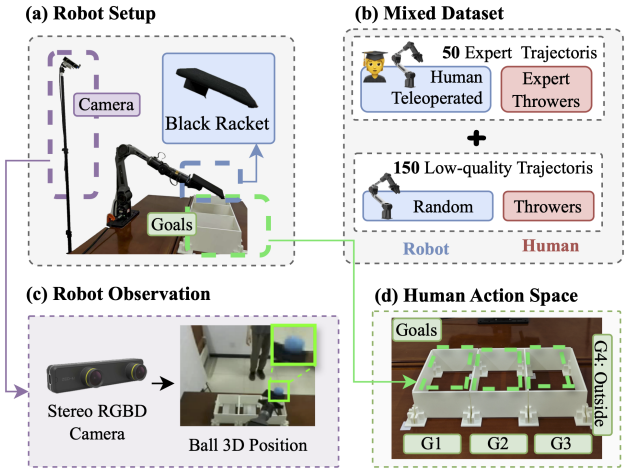


Figure 7: Setups of the human-robot competitive game. G1 to G4 in sub-figure (d) are four actions of the human player.

icy with high exploitability. Corresponding to the design of OFF-FSP, it has the potential to combine with any offline RL algorithms. OFF-FSP with CQL, BCQ, and CRR show similar performances in these two datasets.

5.2 Real-world Human-Robot Confrontation

To further evaluate OFF-FSP, we design a complex real-world zero-sum game—a human-robot ball defence task, which serves as a simplified version of football defense and attack. Figures 1 and 7 illustrate the task scenario. This setup involves a human player, making it difficult to simulate and costly to collect data, thus highlighting the significance of OFF-FSP. The game consists of two players: a human thrower, who attempts to throw a ball into one of three baskets (goals), and a robotic arm acting as a goalkeeper that tries to block the ball. The human thrower receives a reward of $+1$ if the ball enters a goal and -1 otherwise. As a zero-sum game, the robot receives the negative of the human's reward and uses a black racket to defend the goals. The robot is equipped with a stereo RGBD camera that estimates the ball's position in real time (see Figure 7c). Its observation space includes the ball's position and velocity, as well as the position of the robot's end effector. The robot policy outputs target end-effector positions at 60 Hz. These are executed via inverse kinematics and a PD controller. For the human player, we simplify their action space to four discrete targets: the three baskets (G1 to G3) or outside the basket area (G4), as shown in Figure 7d. The target is solely determined by the ball's flight trajectory after release and does not change whether the robot blocks it. Therefore, G4 corresponds to imprecise human throws and represents a poor strategy. The setup of this task is shown in Figure 7.

We collected a mixed dataset comprising 50 expert trajectories and 150 low-quality trajectories. Expert data were collected with a human operator teleoperating the robot arm while another expert performed the throws. The average win rate of the teleoperated robot in these expert trajectories was 64%. Low-quality trajectories were collected by deploying the robot with a random policy, which moves to uniformly sampled positions within a constrained action space. The human thrower was not an expert and occasionally missed the targets (G4). This protocol makes low-quality data cheaper to collect, requiring only a non-expert human thrower.

Methods	G1	G2	G3	Worst	Avg.
BC	0.55	0.35	0.4	0.35	0.43
BC-Expert	0.45	0.5	0.65	0.45	0.53
CQL	0.4	**0.85**	0.8	0.4	0.68
OFF-FSP-CQL	**0.8**	0.75	**0.9**	**0.75**	**0.82**

Table 1: Winning rate of robot in ball defense game. The colored numbers represent the worst case between goals.

Since one player is human, we evaluate only the robot's policy across different algorithms. As *NashConv* is difficult to compute in real-world settings, we adopt the robot's win rate as the evaluation metric. We compare OFF-FSP-CQL with three baselines: BC, BC-Expert, and CQL. Here, BC-Expert denotes BC policy trained only on trajectories with a reward of $+1$. OEF failed in this task, so we removed it from the comparison.

For each goal, a total of 20 throws were performed collectively by 10 human throwers (2 throws per thrower on average). As shown in Table 1, OFF-FSP-CQL outperforms all baselines with consistently robust performance. The dataset shows an imbalanced distribution $P(\text{G1,G2,G3,G4}) = (0.29, 0.11, 0.32, 0.28)$, with relatively fewer throws to the middle goal (G2) and a high proportion of off-target throws (G4). OFF-FSP learns a more balanced human policy $(0.33, 0.22, 0.40, 0.04)$, leading to improved robustness against diverse human strategies. By focusing only on high-reward data, BC-Expert and CQL overfit to average opponents in the dataset and fail to defend consistently across all goal targets (G1–G3).

6 Conclusion

In this paper, we study offline multi-agent reinforcement learning for competitive games and propose OFF-SP and OFF-FSP to enable single-agent offline RL algorithms to be applied in this scenario. We find that OFF-FSP can approximate NE even with partially covered datasets. Extensive and real-world experiments show that all variants of OFF-FSP significantly outperform state-of-the-art baselines in different datasets of multiple two-player zero-sum games.

Acknowledgments

The Shanghai Jiao Tong University team is partially supported by Shanghai Municipal Commission of Economy and Informatization, AI for Science Hundred Teams Hundred Projects, 2025-GZL-RGZN-BTBX-01002. The author, Jingxiao Chen, is supported by the Wu Wen Jun Honorary Doctoral Scholarship, AI Institute, Shanghai Jiao Tong University.

References

Brown, G. W. 1951. Iterative solution of games by fictitious play. *Act. Anal. Prod Allocation*, 13(1): 374.

Brown, N.; and Sandholm, T. 2018. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374): 418–424.

Brown, N.; and Sandholm, T. 2019. Superhuman AI for multiplayer poker. *Science*, 365(6456): 885–890.

Cui, Q.; and Du, S. S. 2022a. Provably efficient offline multi-agent reinforcement learning via strategy-wise bonus. *Advances in Neural Information Processing Systems*, 35: 11739–11751.

Cui, Q.; and Du, S. S. 2022b. When are Offline Two-Player Zero-Sum Markov Games Solvable? *Advances in Neural Information Processing Systems*, 35: 25779–25791.

Fang, F.; Nguyen, T.; Pickles, R.; Lam, W.; Clements, G.; An, B.; Singh, A.; Tambe, M.; and Lemieux, A. 2016. Deploying paws: Field optimization of the protection assistant for wildlife security. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 3966–3973.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, 2052–2062. PMLR.

Gerstgrasser, M.; Trivedi, R.; and Parkes, D. C. 2021. CrowdPlay: Crowdsourcing Human Demonstrations for Offline Learning. In *International Conference on Learning Representations*.

Greenwald, A.; Li, J.; Sodomka, E.; and Littman, M. 2013. Solving for best responses in extensive-form games using reinforcement learning methods. *RLDM 2013*, 116.

Heinrich, J.; Lanctot, M.; and Silver, D. 2015. Fictitious self-play in extensive-form games. In *International conference on machine learning*, 805–813. PMLR.

Hong, Z.-W.; Kumar, A.; Karnik, S.; Bhandwaldar, A.; Srivastava, A.; Pajarinen, J.; Laroche, R.; Gupta, A.; and Agrawal, P. 2023. Beyond uniform sampling: Offline reinforcement learning with imbalanced datasets. *arXiv preprint arXiv:2310.04413*.

Kreps, D. M. 2018. Nash equilibrium. In *The new Palgrave dictionary of economics*, 9251–9258. Springer.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191.

Kurach, K.; Raichuk, A.; Stańczyk, P.; Zając, M.; Bachem, O.; Espeholt, L.; Riquelme, C.; Vincent, D.; Michalski, M.; Bousquet, O.; et al. 2020. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI conference on artificial intelligence*, 4501–4510.

Lanctot, M.; Zambaldi, V.; Gruslys, A.; Lazaridou, A.; Tuyls, K.; Pérolat, J.; Silver, D.; and Graepel, T. 2017. A unified game-theoretic approach to multiagent reinforcement learning. *Advances in neural information processing systems*, 30.

Li, S.; Wang, X.; Zhang, Y.; Cerny, J.; Li, P.; Chan, H.; and An, B. 2022. Offline equilibrium finding. *arXiv preprint arXiv:2207.05285*.

Mathieu, M.; Ozair, S.; Srinivasan, S.; Gulcehre, C.; Zhang, S.; Jiang, R.; Paine, T. L.; Powell, R.; Żołna, K.; Schrittwieser, J.; et al. 2023. AlphaStar Unplugged: Large-Scale Offline Reinforcement Learning. *arXiv preprint arXiv:2308.03526*.

Munos, R.; Stepleton, T.; Harutyunyan, A.; and Bellemare, M. 2016. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems*, 29.

Nachum, O.; Chow, Y.; Dai, B.; and Li, L. 2019. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *Advances in neural information processing systems*, 32.

Qu, Y.; Wang, B.; Shao, J.; Jiang, Y.; Chen, C.; Ye, Z.; Liu, L.; Feng, Y. J.; Lai, L.; Qin, H.; et al. 2023. Hokoff: Real Game Dataset from Honor of Kings and its Offline Reinforcement Learning Benchmarks. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

Silver, D.; and Veness, J. 2010. Monte-Carlo planning in large POMDPs. *Advances in neural information processing systems*, 23.

Timbers, F.; Bard, N.; Lockhart, E.; Lanctot, M.; Schmid, M.; Burch, N.; Schrittwieser, J.; Hubert, T.; and Bowling, M. 2022. Approximate exploitability: learning a best response. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 3487–3493.

Tseng, W.-C.; Wang, T.-H. J.; Lin, Y.-C.; and Isola, P. 2022. Offline Multi-Agent Reinforcement Learning with Knowledge Distillation. *Advances in Neural Information Processing Systems*, 35: 226–237.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354.

Von Stengel, B. 1996. Efficient computation of behavior strategies. *Games and Economic Behavior*, 14(2): 220–246.

Wang, Z.; Novikov, A.; Zolna, K.; Merel, J. S.; Springenberg, J. T.; Reed, S. E.; Shahriari, B.; Siegel, N.; Gulcehre, C.; Heess, N.; et al. 2020. Critic regularized regression.

Advances in Neural Information Processing Systems, 33: 7768–7778.

Wei, H.; Chen, J.; Ji, X.; Qin, H.; Deng, M.; Li, S.; Wang, L.; Zhang, W.; Yu, Y.; Linc, L.; et al. 2022. Honor of kings arena: an environment for generalization in competitive reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 11881–11892.

Yang, R.; Chen, J.; and Narasimhan, K. 2020. Improving dialog systems for negotiation with personality modeling. *arXiv preprint arXiv:2010.09954*.

Yang, Y.; Ma, X.; Li, C.; Zheng, Z.; Zhang, Q.; Huang, G.; Yang, J.; and Zhao, Q. 2021. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 10299–10312.

Ye, D.; Chen, G.; Zhang, W.; Chen, S.; Yuan, B.; Liu, B.; Chen, J.; Liu, Z.; Qiu, F.; Yu, H.; et al. 2020. Towards playing full moba games with deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 621–632.

Ye, D.; Zhang, M.; and Yang, Y. 2015. A multi-agent framework for packet routing in wireless sensor networks. *sensors*, 15(5): 10026–10047.

Yu, C.; Yang, X.; Gao, J.; Chen, J.; Li, Y.; Liu, J.; Xiang, Y.; Huang, R.; Yang, H.; Wu, Y.; et al. 2023. Asynchronous Multi-Agent Reinforcement Learning for Efficient Real-Time Multi-Robot Cooperative Exploration. *arXiv preprint arXiv:2301.03398*.

Yun, W. J.; Park, S.; Kim, J.; Shin, M.; Jung, S.; Mohaisen, D. A.; and Kim, J.-H. 2022. Cooperative multiagent deep reinforcement learning for reliable surveillance via autonomous multi-UAV control. *IEEE Transactions on Industrial Informatics*, 18(10): 7086–7096.

Zhang, R.; Xu, Z.; Ma, C.; Yu, C.; Tu, W.-W.; Tang, W.; Huang, S.; Ye, D.; Ding, W.; Yang, Y.; et al. 2024. A survey on self-play methods in reinforcement learning. *arXiv preprint arXiv:2408.01072*.

Zhong, H.; Xiong, W.; Tan, J.; Wang, L.; Zhang, T.; Wang, Z.; and Yang, Z. 2022. Pessimistic minimax value iteration: Provably efficient equilibrium learning from offline datasets. In *International Conference on Machine Learning*, 27117–27142. PMLR.