Swarm Inverse Reinforcement Learning for Biological Systems

1st Xin Yu
School of Computer Science and Engineering
Beihang University
Beijing, China
nlsdeyuxin@buaa.edu.cn

2nd Wenjun Wu
Institute of Artificial Intelligence
Beihang University
Beijing, China
wwj09315@buaa.edu.cn

3rd Pu Feng
School of Computer Science and Engineering
Beihang University
Beijing, China
fengpu@buaa.edu.cn

4th Yongkai Tian
School of Computer Science and Engineering
Beihang University
Beijing, China
tianyk@buaa.edu.cn

***Abstract*—Complex global behavior can emerge from local interactions in biological systems. Many models have been introduced to describe the interaction rules of biological individuals. Nonetheless, most research efforts cannot capture the inner cognitive and sequential decision process of individual animals in their swarms. In this paper, we formulate this problem as homogeneous Markov game and focus on identifying the potential reward function of individual animals so as to understand their collective behaviors. We propose an inverse reinforcement learning method PS-AIRL specifically for biological systems, where the parameter sharing paradigm is combined with a deep inverse reinforcement learning. Theoretical analysis and experimental evaluation show that PS-AIRL can learn the policy and the reward function from collective behavior demonstrations. Moreover, our methods can be applied to a wide range of biological behavioral studies.**

***Index Terms*—collective intelligence, imitation learning, multi-agent system, reinforcement learning**

I. Introduction

In biological systems, many identical agents interact with each other to achieve a common survival goal and demonstrate collective behavior. Collective behavior can be found in many species such as flocking birds and collective swimming of microorganisms, which bring them advantages over individual behavior including increased diffusion, faster transport, and faster foraging. Various models have been proposed to understand how individual animal decisions based on their local observations can emerge as collective behaviors at the global level. A common conclusion from such research confirms that even simple rules can produce complex behaviors on a global scale [1]. Thus, more investigations must be conducted to reveal the relationship between global behavior and agent-level interactions. Such a research topic is not only a biological interest but also key to cyber-physical system applications such as distributed sensor networks and multi robots systems.

There are two major approaches to study animal interactions in collective behaviors including the agent-based modeling (ABM) and machine learning. ABM is widely used to study complex collective dynamics that consist of autonomous agents interacting with each other [2]. Vicsek model is a seminal agent-based model describing a flocking phase transition. However, constructing such a model needs to propose a set of candidate local rules and compare the results simulated by rules with the desired outcome until an adequate configuration is found. These individual rules are often designed based on domain knowledge and require careful adjustment for parameters [3]. While ABM provides an insightful view of collective behavior, Vicsek models based on the abstraction of particles for individuals cannot capture complex inner cognitive and biophysical factors that are necessary for accurately describing individual behavior [4]. To tackle these issues, machine learning techniques were proposed to model collective behaviors in a data-driven way diminishing the requirement of domain knowledge [5]. For example, deep neural networks were trained to predict the future turning side of a zebrafish and attained higher accuracy than previous agent-based models [6]. Though these models exhibit high accuracy for behavioral prediction, they haven't reveal latent cognitive decision mechanisms to drive individual animals to respond in specific ways towards various swarm behaviors.

A biological system consisting of multiple agents is mainly formulated as Markov game. Markov game assumes that each agent in the system follows a policy aiming to maximize its internal reward. The reward function is a thorough char-

This work was supported in part by the Science and Technology Innovation 2030—"New Generation Artificial Intelligence" Major Project(2018AAA0102300) and the State Key Laboratory of Software Development Environment (Funding No. SKLSDE-2020ZX-01).

acterization of the interacting tendency for the agents [7]. A promising way to understand collective interaction is to recover the potential reward function each agent follows. Inverse reinforcement learning (IRL) provides a data-driven approach to approximate the reward function from collective behavior data. Recent work adopts IRL to find interaction rules in collectives [8], [9]. Unfortunately, these methods only consider the single-agent IRL which is not suitable for collective settings due to the non-stationary environment for individual agents [10]. The biological swarm system is composed of a large number of homogeneous agents and keeps high dimension observation space and action space. Therefore, reconstructing the reward function for biological systems needs to exploiting multi-agent IRL methods. Previous multi-agent IRL methods can't support the scale of biological swarms because of their high computational overhead and poor algorithmic convergence in practice [11]. The remedy to this issue requires considerations for the homogeneity and locality of a biological swarm. Another research efforts in [12] attempt to extend inverse reinforcement learning to homogeneous multi-agent systems. But this algorithm assume simple representations of strategy and reward function, thus not applicable for capturing in high dimensional features in animals' sequential decision mechanisms.

In contrast to previous methods, we propose multi-agent inverse reinforcement learning methods specially adapted to biological systems. Identical individuals interact locally in a biological system contribute to the characteristic of homogeneity and locality. Considering the aforementioned characteristics, we introduce an IRL solution called parameter sharing adversarial inverse reinforcement learning (PS-AIRL) that can solve the high dimensional problem by combining the parameter sharing paradigm and the deep IRL methods. There are two objectives of PS-AIRL: (1) policy imitation, learning policies by imitating biological systems. (2) reward reconstruction, recovering reward functions that induce collective behaviors. We make theoretical analysis and experimental evaluation for the PS-AIRL. The contributions of this paper are summarized as follows:

- By considering homogeneity and locality of biological swarms, we propose a parameter sharing deep IRL method specially adapted to swarm systems.
- We present theoretical analysis for the parameter sharing mechanism in multi-agent reinforcement learning to illustrate the effectiveness of the PS-AIRL framework.
- We conduct a variety of experiments in two major swarm scenarios. Experimental results show that the PS-AIRL can explain and accurately reproduce the collective behavior of a swarm system.

II. Related work

A. Agent-based models

Modeling the behavior of the collective biological systems is related directly or indirectly to mathematical agent-based models. Construction of these mathematical models often follows a cycle paradigm [3]. First, based on prior knowledge, researchers need to design a set of candidate local rules to simulate the collective behavior numerically. Second, they compare the simulation results with observation of animal behavior, and refine the rules until they find an adequate collective configuration. This cycle paradigm is time-consuming and requires considerable domain knowledge. For example, the Vicsek model regards biological swarm systems as self-propelled particles with a tendency to align their motion directions with their neighbors. As an extension of the Vicsek model, the Cucker-Smale model [13] specifies that all agents determine their next velocity by calculating the weighted average of the speed difference of their neighbors. In the context of fish schooling, agent-based models have generally been constructed by assuming simple behavioral rules, such as the popular "3-A rules" of avoidance, alignment, and attraction [14]. However, there often exists inconsistency between such an agent-based model and the actual behavior due to the specificity of a species swarm behavior. It is well known that different species have different behavior patterns. Given the unavoidable simplification in a universal common model, it is highly unlikely that biologists can create a single agent-based model for all the species swarming behavior [15].

B. Machine learning methods

To tackle the challenge brought by ABM, many attempts have been made to incorporate machine learning (ML) techniques and empirical data into agent-based models [5]. Benefit from the rapid growth of deep learning, researchers using deep neural networks to model collective behavior [16]. Researcher trained a neural network to predict the future turning side of a zebrafish in a collective [6]. By analyzing the different models, they obtain insight into how fish interact and how they aggregate information from different neighbors. To increase the model accuracy, a graph neural network was devised to imitate the behavior of an observed swarm of agents [17]. Though these models predict the behavior accurately, the potential motivation for animal behavior are not yet fully understood.

C. Inverse reinforcement learning

By formulating the collective behavior as a Markov game, we can recover the potential reward function of each agent to understand the decision process for collective interaction. Inverse reinforcement learning implements such a process of inferring a reward function from observed behavior [18]. The most widely used inverse reinforcement learning algorithm includes generative adversarial imitation learning (GAIL) and adversarial inverse reinforcement learning (AIRL), both of which are based on the generative adversarial framework [19] [20]. Bayesian Inverse Reinforcement Learning has been applied to learn the local rules governing collective movement for a captive guppy population [21]. The leader-follower network in the flocks of birds can be inferred by the single-agent IRL method [8]. However, the single-agent IRL methods are insufficient in solving the collective scenarios and none of them consider the characteristic of biological collective systems. The Multi-agent adversarial inverse reinforcement learning (MA-AIRL) proposed on the general Markov game shows promising prospects, which can recover the reward function from multi-agent systems in a very small scale [22]. But for biological swarms with large-scale individuals, these general multi-agent IRL methods are not able to deliver convergence results. To our best knowledge, the most relevant algorithm with our work is the swarm inverse reinforcement learning proposed in [12]. It reduces the swarm IRL problem to a single-agent IRL problem, thus achieving more efficient computation for swarms with simple decisive strategies. Its major limitation lies upon its shallow representation of swarm agents. Without deep neural networks for modelling each agent's cognitive process, it can't describe complex behaviors of various biological swarm systems.

III. Backgrounds

This section presents the notation definition and a brief introduction to adversarial inverse reinforcement learning adopted in our framework.

A. Problem formulation

A biological swarm system often exhibits the characteristics of homogeneity and locality.

- Homogeneity: All agents in the system carry a common architecture (i.e. The same observation space and action space)
- Locality: The agents can observe only parts of the system within a certain range. Their decisions depend on their current neighborhood only.

In principle, any system with these properties can be described as a Markov game [23]. Given the homogeneity property of swarm systems, we can further adopt the swar-MDPs that explicitly implements a homogeneous Markov game architecture [12]. The SwarMDP framework is defined as a tuple $(N, S, O, A, R, T, \pi, \xi)$.

- N is the number of agents in the system.
- S, O, A are sets of local states,observations and actions respectively.
- $R : O \to R$ is an agent-level reward function.
- $T : S^N \times A^N \times S^N \to R$ is the global transition model of the system. The system reaches state $\tilde{s} = (\tilde{s}^{(1)}, \ldots, \tilde{s}^{(N)})$when the agents perform the joint action $a = (a^{(1)}, \ldots, a^{(N)})$ at state as $T(\tilde{s}|s, a)$,where $s^{(n)}, \tilde{s}^{(n)} \in S$ and $a^{(n)} \in A_1$ represent the local states and the local action of agent n, respectively.
- $\pi : O \to A$ is the local policy.
- $\xi : S^N \to O^N$ is the observation model of the system.

The observation model ξ indicates every agent's sensing capability, which defines their perception of a given system state $s \in S^N$. $\xi(s) = (\xi^{(1)}(s), \ldots, \xi^{(N)}(s)) \in O^N$ denotes the ordered set of local observations passed on to the agents at state s. The agent n have no access to their local states $s^{(n)} \in S$ but only to their local observations $o^n = \xi^{(n)}(s) \in O$. For example, in a flocking of birds, $\xi^{(n)}$ could represent a bird's local perception of its immediate neighbors.

B. Adversarial inverse reinforcement learning

Adversarial Inverse Reinforcement Learning (AIRL) is a promising inverse reinforcement learning method based on the generative adversarial framework which consists of the generator and the discriminator [24]. The goal of AIRL is to reconstruct the expert's policy and reward function from demonstration τ_E. In AIRL, the agent interacts with the environment by executing a policy network π to generate trajectories τ_j. And these trajectories τ_j must be compared against the expert trajectories τ_E by the discriminator D_ω parameterised by ω. The discriminator probability of AIRL takes on a particular form:

$$D_\omega(s, a) = \frac{\exp(f_\omega(s, a))}{\exp(f_\omega(s, a)) + \pi(a|s)} \tag{1}$$

where $f_\omega(s, a)$ is a learned function. The policy π is trained to maximize $R(s, a) = \log(1 - D_\omega) - \log D_\omega$. To tackle the reward shaping ambiguity, where many reward functions can explain an optimal policy, the function f was restricted to a reward estimator g and a potential shaping function h by (2). It has been shown that under suitable assumptions, g_ω and h_ϕ can recover the true reward function.

$$f_{\omega,\phi}(s, a, s') = g_\omega(s, a) + \gamma h_\phi(s') - h_\phi(s) \tag{2}$$

C. Policy gradient and Actor-Critic

Policy Gradient (PG) algorithms are a class of model-free RL algorithms that aim to directly learn a policy to maximize the expected returns [25]. Proximal Policy Optimization (PPO) is an Actor-Critic method. It can be used to train the policy network in the AIRL. Actor-critic (AC) algorithms estimate returns using a value function $V_\pi(s; \theta)$ with parameters θ. In the AC algorithm, the neural network approximates the policy function and value function. In a multi-agent partially observable setting, the simplest AC algorithm defines a policy loss $L(\phi_i)$ and a value loss $L(\theta_i)$ for agent i.

$$L(\phi_i) = -\log \pi(a_t^i | o_t^i; \phi_i)((r_t^i + \gamma V(o_{t+1}^i; \theta_i) - V(o_t^i; \theta_i)) \tag{3}$$

$$L(\theta_i) = \|V(o_t^i; \theta_i) - y_i\|^2 \text{ with } y_i = r_t^i + \gamma V(o_{t+1}^i; \theta_i) \tag{4}$$

IV. Methods

This section describes our swarm inverse reinforcement learning method PS-AIRL in detail. It combines the parameter sharing paradigm and a deep IRL algorithm to study biological swarm behaviors.

A. Parameter sharing for biological system

The number of animals in a biological swarm system is often very large. To efficiently train policy networks for such a large homogeneous multi-agent swarm, it is common to adopt parameter sharing technique, where all agents share the same parameter in their policy networks [26]. Though widely used, the effectiveness of the parameter sharing paradigm has not been confirmed by theoretical analysis and is often thought of as an implementation detail [27].

From the perspective of one agent, other agents are perceived as part of the environment. In each episode, every agent interacts with the environment to generate its trajectories. Traditionally, an agent learns its policy network and value network from its own experience according to the loss functions specified in (3) and (4) respectively. To accelerate training speed, Christianos et al. [28] applies experience sharing by combining the gradients of different agents to update the policy separately. Each agent is trained from experience generated from different policies, inducing more diverse explorations. Their main theoretical results are displayed in equation (5) and (6) indicating that the policy network and the value network can be updated by their own trajectories as well as the experience of other agents in a multi-agent scenario. The hyperparameter λ weights the experience of other agents.

$$\begin{aligned} L(\phi_i) =& -\log \pi(a_t^i|o_t^i;\phi_i)(r_t^i+\gamma V(o_{t+1}^i;\theta_i)-V(o_t^i;\theta_i)) \\ & -\lambda \sum_{k\neq i} \frac{\pi(a_t^k|o_t^k;\phi_i)}{\pi(a_t^k|o_t^k;\phi_k)} \log \pi(a_t^k|o_t^k;\phi_i)\cdot \\ & (r_t^k+\gamma V(o_{t+1}^k \theta_i)-V(o_t^k;\theta_i)) \end{aligned} \tag{5}$$

$$\begin{aligned} L(\theta_i) =& \|V(o_t^i;\theta_i)-y_i^i\|^2 + \lambda \sum_{k\neq i} \frac{\pi(a_t^k|o_t^k;\phi_i)}{\pi(a_t^k|o_t^k;\phi_k)} \cdot \\ & \|V(o_t^k;\theta_i)-y_k^i\|^2 \end{aligned} \tag{6}$$

The intuition of parameter sharing is that all the agents share the same network, which is learned from all the trajectories. All the agents execute the same policy illustrated in equation(7).

$$\pi(a_t^k|o_t^k;\phi_i) = \pi(a_t^k|o_t^k;\phi_k) \tag{7}$$

The loss function (5) and (6) of the policy network and the value network are reduced to (8) and (9), which means that the policy network and the value network can be updated by all the trajectories generated from the biological system.

$$L(\phi_i) = -\sum_k \log \pi(a_t^k|o_t^k;\phi_i)(r_t^k+\gamma V(o_{t+1}^k;\theta_i)-V(o_t^k;\theta_i)) \tag{8}$$

$$L(\theta_i) = \sum_k \|V(o_t^k;\theta_i)-y_k^i\|^2 \tag{9}$$

During the reinforcement learning process in the environment, each agent executes the same policy network. Equation (8) and (9) shows that the sum of policy and value loss gradients are used to optimise the shared parameters.

B. Parameter sharing AIRL

Fig. 1 shows the PS-AIRL framework, where we formulate biological swarm systems by swarMDP and extend the AIRL by sharing the generator and discriminator among all the agents. In swarm imitation learning settings, we do not have access to the reward function, but have demonstrations provided by experts (N expert agents in swarMDP). The goal of PS-AIRL is to infer the right reward function for the agents so as to mimic the experts' policies π^E.

The learning procedure for PS-AIRL is summarized in Algorithm 1. The demonstrations can be denoted as $\tau_E = \{\tau_j\}_{j=1}^M$, where $\tau_j = \{(s_j^t, \boldsymbol{a}_j^t)\}_{t=1}^T$ is the trajectory of animal j collected from step 1 to step T. In the first step, we maintain a collective demonstrations τ_E collected by $a_t = \pi_E(a_t|s_t)$. PS-AIRL randomly initializes the policy network π and the discriminator network $D_{\theta,\phi}$. Each agent interacts with the environment independently according to the current policy. The resulting trajectories denoted as $\vec{\tau}$ are sampled by different agents following the same policy π_{θ_k}. The training process of discriminator $D_{\theta,\phi}$ in multi-agent IRL problem is the same as single-agent one. PS-AIRL trains $D_{\theta,\phi}$ via binary logistic regression to classify expert data τ_E from samples $\vec{\tau}$. After training the discriminator, it can update the reward function(Algorithm 1, Line 6). In each iteration, the algorithm trains the policy network π by Algorithm 2.

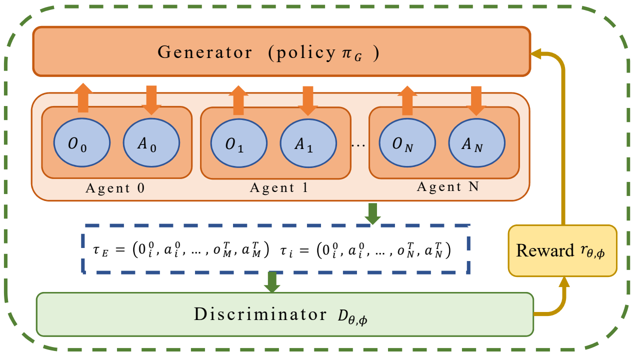
Generator (policy π_G)
O_0
A_0
Agent 0
O_1
A_1
Agent 1
O_N
A_N
Agent N
$\tau_E = (0_i^0, a_i^0, \dots, o_M^T, a_M^T)$ $\tau_i = (0_i^0, a_i^0, \dots, o_N^T, a_N^T)$
Reward $r_{\theta,\phi}$
Discriminator $D_{\theta,\phi}$

Fig. 1. PS-AIRL framework for biological system.

Algorithm 1 PS-AIRL

```
1: Input: expert trajectories$\tau_E \sim \pi_E$
2: initialize policy $\pi_{\theta_0}$ and discriminator$D_{\theta_0,\phi_0}$
3: for $k \leftarrow 1, 2, \dots$ do
4:     Rollout trajectories for all agents $\vec{\tau} \sim \pi_{\theta_k}$
5:     Train $D_{\theta_k,\phi_k}$ via binary logistic regression to classify
       expert data $\tau_E$ from samples $\vec{\tau}$.
6:     Generating reward $r_{\theta_k,\phi_k}$
           $r_{\theta_k,\phi_k} \to \log D_{\theta_k,\phi_k} - \log(1 - D_{\theta_k,\phi_k})$
7:     Updating policy $\pi_k$ with respect to $r_{\theta_k,\phi_k}$ by PS-PPO
8: end for
```

As shown in Algorithm 2 we extend proximal policy optimization algorithms (PPO) to multi-agent setting by parameter

Algorithm 2 PS-PPO

```
1: Initialize policy network$\pi_\theta$ and value function $V_\phi$
2: for $k = 1$ to $M$ do
3:     Agents j=0,1,2,3,...M execute policy $\pi_{\theta_k}(o_j)$,collect trajectories $D_k = \tau_i{}^j$
4:     Trajectories data normalization
5:     Calculate the accumulated discount reward $\widehat{R}_t$
6:     Estimate advantage function $\hat{A}_t$ based on the current value function
7:     Update the policy
           $\theta_{k+1} = \arg\max_\theta \frac{1}{|D_k|T} \sum_{\tau\in D_k} \sum_{t=0}^{T} \min (\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t,a_t), g(\epsilon, A^{\pi_{\theta_k}}(s_t,a_t)))$
8:     Update the value function
           $\phi_{k+1} = \arg\min_\phi \frac{1}{|D_k|T} \sum_{\tau\in D_k} \sum_{t=0}^{T} (V_\phi(s_t) - \hat{R}_t)^2$
9: end for
```

sharing [29]. At each iteration of the PS-PPO, each agent collects samples according to $\pi_{\theta_k}(o_j)$. After that, we aggregate the samples of all the agents as a batch of data to calculate the advantage function and update network parameters.

V. Experiment and Discussion

In this section, we evaluate the performance of the PS-AIRL algorithm in the following two scenarios. The first scenario is conducted in Rendezvous Problem [30], where the goal is to minimize the distances between all the agents. The expert agents were trained by reinforcement learning algorithms using the true reward function. The second scenario is to align each agent's movement direction. The Vicsek model was used to construct the expert policy. We seek to recover both the policy function and the reward function from the expert demonstration. We evaluated AIRL under the setting of different agent numbers as well as different quantities of expert demonstration. Moreover, We compare our methods with behavioral cloning (BC), which learns a maximum likelihood estimate for a_i given each state s and does not require actions from other agents [31].

A. Environment details

We define the state of an agent by $s^i = [x^i, y^i, \phi^i]$. Specifically, $0 \le x \le x_{\max}, 0 \le y \le y_{\max}, 0 \le \phi < 2\pi$. The action of an agent is defined as linear velocity v and angular velocity ω. The kinematic model is given by(10). We include more details about the implementation of our work in the Appendix.

$$\begin{aligned}\dot{x} &= v\cos\phi \\ \dot{y} &= v\sin\phi \\ \dot{\phi} &= \omega.\end{aligned} \tag{10}$$

B. Rendezvous Problem

The ground truth of the reward function in this scenario is defined in (11), in which the reward value increases when the distances between all the agents decrease. Following this reward specification, We train the expert agents based on PS-PPO. In this process, the resultant policy π_E is constructed to generate the expert demonstration τ_E. Then we use the PS-AIRL to imitate the expert policy and reconstruct the reward function.

$$R(s) = \sum_{i=1}^{N} \sum_{j=i+1}^{N} d^{i,j} \tag{11}$$

Imitation performance of different algorithms are evaluated via the true accumulated reward obtained in an episode. Fig. 2 displays the performance of PS-AIRL, Expert, BC and Random policy. Naturally, the performance of BC increases with more expert demonstrations. PS-AIRL performs consistently better than BC in all the settings. It can be seen from the results that PS-AIRL is not sensitive to the increase of the number of agents. In fact, the parameter sharing mechanism introduced on the basis of the homogeneity assumption makes PS-AIRL is less prone to changes in the number of agents. Table I shows the imitation evaluation results under different number of agents and different number of expert trajectories. It further confirms that PS-AIRL outperforms the other algorithms can achieve a performance that are close to the experts.

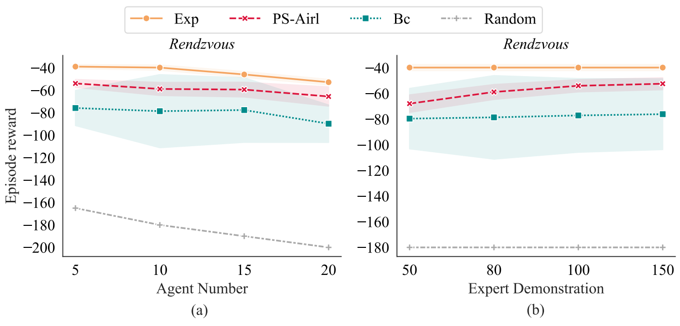

Fig. 2. Episode rewards of the imitation policy in the Rendezvous problem. (a) Episode reward of imitating collective systems with different agent numbers. (b)Episode rewards of imitating collective policy trained by different expert demonstrations

We interpret the ability of reward learning of PS-AIRL by visualizing the reward function. We sample an action set from the action space and calculate the reward for each action in a particular scenario. Fig. 2 visualizes the reward function in a time step. Fig. 2(a) is the visualization of the reward function for the focal agent in Fig. 2(b). The horizontal and vertical coordinates are the linear velocity and angular velocity of the agent. The blue color represents the smaller reward value than red color. In the Fig. 2(b), the target agent is driving away from other agents. Fig. 2(a) shows that in the current state, the linear velocity has a small effect on the focal agents' reward, and the focal agent tends to have a higher angular velocity. This observation indicates that the agent will get higher reward value when it turns clockwise or counterclockwise to other

TABLE I
POLICY IMITATING PERFORMANCE IN RENDEZVOUS TASKS

#Agents	Algorithm	Expert Trajectories			
		50	80	100	150
5	Expert	-38.86±1.85			
	PS-AIRL	-56.78±4.48	-53.86±3.56	-52.34±3.34	-52.28±2.88
	Behavioral cloning	-78.83±16.67	-75.76±15.67	-68.68±17.86	-67.86±14.56
10	Expert	-39.74±2.36			
	PS-AIRL	-67.86±6.84	-58.78±5.83	-53.95±4.78	-52.35±4.68
	Behavioral cloning	-79.56±23.56	-78.57±32.58.	-77.07±28.73	-76.08±27.65
15	Expert	-45.86±2.64			
	PS-AIRL	-68.24±8.58	-59.34±6.57	-57.43±5.87	-56.38±4.65
	Behavioral cloning	-70.34±24.67	-77.57±28.78	-76.87±24.58	-85.78±23.57
20	Expert	-52.76±2.56			
	PS-AIRL	-69.32±9.64	-65.48±8.58	-63.24±7.69	-62.76±5.78
	Behavioral cloning	-92.65±18.78	-89.75±16.69	-90.56±15.78	-88.65±13.87

agents. This behavior is exactly what is needed to complete the rendezvous goal. The experimental results show that the swarm inverse reinforcement learning algorithm based on PS-AIRL can identify a reasonable reward function.

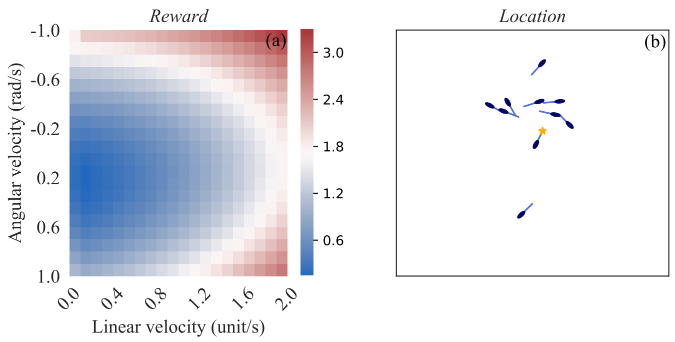

Fig. 3. Visualization of the learned reward function for the rendezvous agents. (a)The reward under different action for the focal agent.(b)The focal agent is denoted by yellow star.

C. Vicsek model

We test the PS-AIRL framework on the demonstration data generated by the Vicsek model [2]. The model consists of a fixed number of particles, living in the unit square with periodic boundary conditions. Each agent n moves with a constant absolute velocity v and is characterized by its location $x_t^{(n)}$ and orientation $\theta_t^{(n)}$ in the plane. The neighborhood of the agents is determined by interaction radius ρ. At each time instance, the agents' orientations get synchronously updated to the average orientation of their neighbors (including themselves) with additive random perturbations $\Delta\theta_t^{(n)}$. Equation (12) describes the details of the Vicsek model. $\langle\theta_t^{(n)}\rangle_\rho$ denotes the mean orientation of all the agents within the ρ neighborhood of agent n at time t.

$$\begin{aligned} \theta_{t+1}^{(n)} &= \langle\theta_t^{(n)}\rangle_\rho + \Delta\theta_t^{(n)} \\ x_{t+1}^{(n)} &= x_t^{(n)} + v_t^{(n)} \end{aligned} \tag{12}$$

Our goal is to learn a model for this expert behavior from recorded agent trajectories using the proposed framework. We define the order parameter in(13)to evaluate the imitating policy behavior.

$$\omega_t = \frac{1}{Nv}\left|\sum_{n=1}^{N} v_t^{(n)}\right| \in [0,1] \tag{13}$$

The overall task performance is evaluated based on their cumulative order parameters. Each episode is fixed 300 time steps. Fig. 4 shows the performance of the expert strategy, PS-AIRL, BC and random strategy. It can be seen from the results that PS-AIRL is not sensitive to the increase in the number of agents, and can achieve a performance that are close to the experts. Table II shows the imitation evaluation results under different number of agents and different number of expert trajectories.

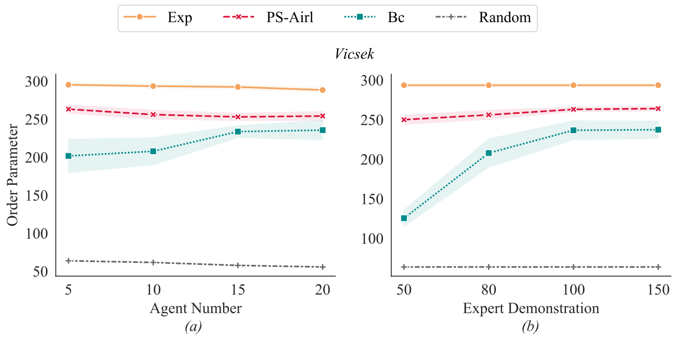

Fig. 4. Episode Order parameter of the imitation policy for the Vicsek model. (a) Episode Order parameter of imitating collective systems with different agent numbers. (b)Episode Order parameter of imitating collective policy trained by different expert demonstrations

The Fig. 5 visualizes the reward function of the Vicsek model. The horizontal and vertical coordinates of the Fig. 5(a) are the linear velocity and angular velocity of the agent. The average motion direction of the agents is shown by the red arrow in Fig. 5(a). There is a certain gap between the direction of the focal agent and the average direction, the focal agent needs to rotate anticlockwise to meet the motion rules of the Vicsek model. The Fig. 5(a) displays that when the agent rotates anticlockwise, it will obtain a higher reward. Although

TABLE II
POLICY IMITATING PERFORMANCE FOR VICSEK MODEL

#Agents	Algorithm	Expert Trajectories			
		50	80	100	150
5	Expert	295.32±1.85			
	PS-AIRL	259.06±5.61	263.25±5.31	266.29±2.46	267.30±2.47
	Behavioral cloning	164.11±31.23	201.70±21.72	204.51±16.37	206.51±13.37
10	Expert	293.53±2.39			
	PS-AIRL	249.96±5.59	256.07±5.58	263.01±2.67	264.03±2.59
	Behavioral cloning	125.37±10.14	207.81±17.72	236.59±11.94	237.31±10.85
15	Expert	292.41±2.73			
	PS-AIRL	253.46±7.21	253.03±4.59	257.12±6.20	258.13±6.15
	Behavioral cloning	173.60±10.42	233.68±7.62	244.55±5.98	245.63±5.96
20	Expert	288.42±2.85			
	PS-AIRL	249.54±2.49	254.19±5.72	260.22±3.87	262.77±5.54
	Behavioral cloning	180.53±15.53	235.63±12.62	220.55±12.58	243.62±13.62

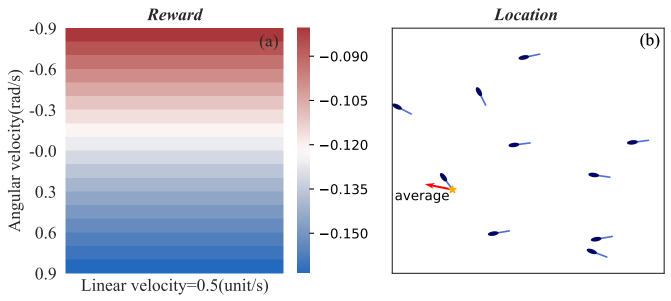

Fig. 5. Visualization of the learned reward function for the Vicsek model. (a)The reward under different actions for the focal agent.(b) The orientation alignment of ten moving agents. The focal agent is denoted by yellow star.

there is no ”true” reward model for the Vicsek system, one can see from the system equations in(12) that the agents tend to align over time. The experimental results show that PS-AIRL can reasonably identify the reward function of the Vicsek model.

VI. DISCUSSION AND FUTURE WORK

We propose a swarm inverse reinforcement learning method called PS-AIRL specifically for collective biological systems. Biological systems are often composed of many agents having large action and observation spaces, challenging for physical models to discover and interpret inner cognition mechanisms. Though the traditional models predict the behaviors, the latent cognitive process for animal collective behaviors is not yet fully understood. PS-AIRL can imitate the expert policy and reconstruct the reward function based on demonstration data. We conduct a theoretical analysis of the parameter-sharing paradigm showing that it can be used to extend the single inverse reinforcement learning to a collective setting in the biological system. Experimental results show that PS-AIRL can achieve excellent performance close to the expert system and reconstruct an explainable reward function for the agents.

Based on the current work on PS-AIRL, we plan to study reward function and strategy function of different specifies on more swarming scenarios. More thorough analysis and interpretation of the inferred reward function should be done to study the relationship between individual interactions and group behavior.

ACKNOWLEDGMENT

We gratefully acknowledge the support of the Science and Technology Innovation 2030—“New Generation Artificial Intelligence” Major Project (Funding No. 2018AAA0102300) and the State Key Laboratory of Software Development Environment (Funding No. SKLSDE-2020ZX-01).

REFERENCES

[1] I. D. Couzin, J. Krause, R. James, G. D. Ruxton, and N. R. Franks, ”Collective memory and spatial sorting in animal groups,” Journal of theoretical biology, vol. 218, no. 1, pp. 1-11, 2002.
[2] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, ”Novel type of phase transition in a system of self-driven particles,” Physical review letters, vol. 75, no. 6, p. 1226, 1995.
[3] D. J. Sumpter, R. P. Mann, and A. Perna, ”The modelling cycle for collective animal behaviour,” Interface focus, vol. 2, no. 6, pp. 764-773, 2012.
[4] K. Ried, T. Müller, and H. J. Briegel, ”Modelling collective motion based on the principle of agency: General framework and the case of marching locusts,” PloS one, vol. 14, no. 2, p. e0212044, 2019.
[5] P. Torrens, X. Li, and W. A. Griffin, ”Building agent-based walking models by machine-learning on diverse databases of space-time trajectory samples,” Transactions in GIS, vol. 15, pp. 67-94, 2011. .
[6] F. J. H. Heras, F. Romero-Ferrero, R. C. Hinz, and G. G. de Polavieja, ”Deep attention networks reveal the rules of collective motion in zebrafish,” PLoS Comput Biol, vol. 15, no. 9, p. e1007354, Sep 2019.
[7] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, ”An algorithmic perspective on imitation learning,” Foundations and Trends in Robotics, vol. 7, no. 1-2, pp. 1-179, 2018.
[8] R. Pinsler, M. Maag, O. Arenz, and G. Neumann, ”Inverse reinforcement learning of bird flocking behavior,” in ICRA Swarms Workshop, 2018.
[9] S. Yamaguchi et al., ”Identification of animal behavioral strategies by inverse reinforcement learning,” PLoS computational biology, vol. 14, no. 5, p. e1006122, 2018.
[10] G. Papoudakis, F. Christianos, A. Rahman, and S. V. Albrecht, ”Dealing with non-stationarity in multi-agent deep reinforcement learning,” arXiv preprint arXiv:1906.04737, 2019.
[11] J. Song, H. Ren, D. Sadigh, and S. Ermon, ”Multi-agent generative adversarial imitation learning,” in Advances in neural information processing systems, 2018, pp. 7461-7472.
[12] A. Šošić, W. R. KhudaBukhsh, A. M. Zoubir, and H. Koeppl, ”Inverse Reinforcement Learning in Swarm Systems,” in Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, 2017, pp. 1413-1421.
[13] F. Cucker and S. Smale, ”Emergent behavior in flocks,” IEEE Transactions on automatic control, vol. 52, no. 5, pp. 852-862, 2007.

[14] J. E. Herbert-Read, A. Perna, R. P. Mann, T. M. Schaerf, D. J. Sumpter, and A. J. Ward, "Inferring the rules of interaction of shoaling fish," Proceedings of the National Academy of Sciences, vol. 108, no. 46, pp. 18726-18731, 2011.

[15] N. T. Ouellette and D. M. Gordon, "Goals and Limitations of Modeling Collective Behavior in Biological Systems," Frontiers in Physics, vol. 9, p. 341, 2021.

[16] R. Bastien and P. Romanczuk, "A model of collective behavior based purely on vision," Science advances, vol. 6, no. 6, p. eaay0792, 2020.

[17] S. Zhou, M. J. Phielipp, J. A. Sefair, S. I. Walker, and H. B. Amor, "Clone swarms: Learning to predict and control multi-robot systems by imitation," in 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019: IEEE, pp. 4092-4099.

[18] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in Proceedings of the twenty-first international conference on Machine learning, 2004, p. 1.

[19] J. Ho and S. Ermon, "Generative adversarial imitation learning," in Advances in neural information processing systems, 2016, pp. 4565-4573.

[20] J. Fu, K. Luo, and S. Levine, "Learning Robust Rewards with Adverserial Inverse Reinforcement Learning," in International Conference on Learning Representations, 2018.

[21] T. L. Schafer, C. K. Wikle, and M. B. Hooten, "Bayesian Inverse Reinforcement Learning for Collective Animal Movement," arXiv preprint arXiv:2009.04003, 2020.

[22] L. Yu, J. Song, and S. Ermon, "Multi-agent adversarial inverse reinforcement learning," in International Conference on Machine Learning, 2019: PMLR, pp. 7194-7201.

[23] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in Machine learning proceedings 1994: Elsevier, 1994, pp. 157-163.

[24] I. J. Goodfellow et al., "Generative Adversarial Networks," arXiv e-prints, p. arXiv:1406.2661. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2014arXiv1406.2661G

[25] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in Advances in neural information processing systems, 2000, pp. 1057-1063.

[26] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in International Conference on Autonomous Agents and Multiagent Systems, 2017: Springer, pp. 66-83.

[27] F. Christianos, G. Papoudakis, A. Rahman, and S. V. Albrecht, "Scaling Multi-Agent Reinforcement Learning with Selective Parameter Sharing," in International Conference on Machine Learning, 2021: PMLR, pp. 1989-1998.

[28] F. Christianos, L. Schäfer, and S. V. Albrecht, "Shared Experience Actor-Critic for Multi-Agent Reinforcement Learning," in Thirty-fourth Conference on Neural Information Processing Systems, 2020: Curran Associates Inc, pp. 10707-10717.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," p. arXiv:1707.06347. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2017arXiv170706347S

[30] M. Hüttenrauch, S. Adrian, and G. Neumann, "Deep reinforcement learning for swarm systems," Journal of Machine Learning Research, vol. 20, no. 54, pp. 1-31, 2019.

[31] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," Neural computation, vol. 3, no. 1, pp. 88-97, 1991.

Appendix

A. Observation Model

Considering the locality of the biological system, the observation model converts the state of the system to the agents' observation. Fig. 6 shows the agent's observation model, where an agent i can sense the following properties about other agents $j \in N(i)$ within its neighborhood:

$$\begin{aligned} &d^{i,j} \\ &\phi^{i,j} = \arctan\left(\frac{y^j - y^i}{x^j - x^i}\right) - \phi^i \\ &\theta^{i,j} = \arctan\left(\frac{y^i - y^j}{x^i - x^j}\right) - \phi^j \\ &\Delta v^{i,j} = v^i \left[\cos\phi^i, \sin\phi^i\right] - v^j \left[\cos\phi^j, \sin\phi^j\right] \end{aligned} \tag{14}$$

$d_{i,j}, \phi^{i,j}, \theta^{i,j}, \Delta v^{i,j}$ denotes distance to neighboring agents, bearing to neighboring agents, relative orientation, relative velocity respectively.

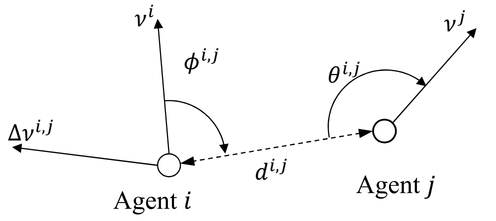

Fig. 6. Illustration of two neighboring agents facing the direction of their velocity vectorsv^iandv^j, along with the observed quantities.

B. Network architecture

The observation information $o_{i,j}$ that agent i receives of agent j is composed of $d_{i,j}, \phi^{i,j}, \theta^{i,j}, \Delta v^{i,j}$. We concatenate the quantities $\{o^{i,j}\}_{j\in N(i)}$ into a single observation vector O_i.

The policy network maps the observation vector O_i to the action a. We process the observation vector with two hidden layer of 64 neurons. The output of the second layer is mapped to the action. Fig. 7 shows our policy network architecture.

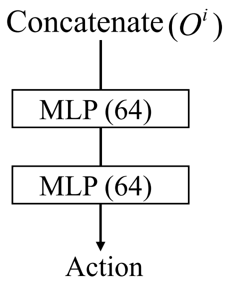

Fig. 7. Policy network architecture.

As for the discriminator network, we use a linear function approximator for the reward term g and a 2-layer MLP network for the shaping term h.