Inverse Reinforcement Learning in Swarm Systems

Adrian Šošić, Wasiur R. KhudaBukhsh, Abdelhak M. Zoubir and Heinz Koeppl

Department of Electrical Engineering and Information Technology
Technische Universität Darmstadt, Germany

ABSTRACT

Inverse reinforcement learning (IRL) has become a useful tool for learning behavioral models from demonstration data. However, IRL remains mostly unexplored for multi-agent systems. In this paper, we show how the principle of IRL can be extended to homogeneous large-scale problems, inspired by the collective swarming behavior of natural systems. In particular, we make the following contributions to the field: 1) We introduce the swarMDP framework, a subclass of decentralized partially observable Markov decision processes endowed with a swarm characterization. 2) Exploiting the inherent homogeneity of this framework, we reduce the resulting multi-agent IRL problem to a single-agent one by proving that the agent-specific value functions in this model coincide. 3) To solve the corresponding control problem, we propose a novel heterogeneous learning scheme that is particularly tailored to the swarm setting. Results on two example systems demonstrate that our framework is able to produce meaningful local reward models from which we can replicate the observed global system dynamics.

Keywords

inverse reinforcement learning; multi-agent systems; swarms

1. INTRODUCTION

Emergence and the ability of self-organization are fascinating characteristics of natural systems with interacting agents. Without a central controller, these systems are inherently robust to failure while, at the same time, they show remarkable large-scale dynamics that allow for fast adaptation to changing environments [5, 6]. Interestingly, for large system sizes, it is often not the complexity of the individual agent, but the (local) coupling of the agents that predominantly governs the final system dynamics. It has been shown [22, 31], in fact, that even relatively simple local dynamics can result in various kinds of higher-order complexity at a global scale when coupled through a network with many agents. Unfortunately, the complex relationship between the global behavior of a system and its local implementation at the agent level is not well understood. In particular, it remains unclear when – and how – a global system objective can be encoded in terms of local rules, and what are the requirements on the complexity of the individual agent in order for the collective to fulfil a certain task. Yet, this understanding is key to many of today's and future applications, such as distributed sensor networks [13], nanomedicine [8], programmable matter [9], and self-assembly systems [33].

A promising concept to fill this missing link is inverse reinforcement learning (IRL), which provides a data-driven framework for learning behavioral models from expert systems [34]. In the past, IRL has been applied successfully in many disciplines and the learned models were reported to even outperform the expert system in several cases [1, 16, 27]. Unfortunately, IRL is mostly unexplored for multi-agent systems; in fact, there exist only few models which transfer the concept of IRL to systems with more than one agent. One such example is the work presented in [18], where the authors extended the IRL principle to non-cooperative multi-agent problems in order to learn a joint reward model that is able to explain the system behavior at a global scale. However, the authors assume that all agents in the network are controlled by a central mediator, an assumption which is clearly inappropriate for self-organizing systems. A decentralized solution was later presented in [25] but the proposed algorithm is based on the simplifying assumption that all agents are informed about the global state of the system. Finally, the authors of [7] presented a multi-agent framework based on mechanism design, which can be used to refine a given reward model in order to promote a certain system behavior. However, the framework is not able to learn the reward structure entirely from demonstration data.

In contrast to previous work on multi-agent IRL, we do *not* aspire to find a general solution for the entire class of multi-agent systems; instead, we focus on the important subclass of homogeneous systems or *swarms*. Motivated by the above-mentioned questions, we present a scalable IRL solution for the swarm setting to learn a single *local* reward function which explains the *global* behavior of a swarm, and which can be used to reconstruct this behavior from local interactions at the agent level. In particular, we make the following contributions: 1) We introduce the *swarMDP*, a formal framework to compactly describe homogeneous multi-agent control problems. 2) Exploiting the inherent homogeneity of this framework, we show that the resulting IRL problem can be effectively reduced to the single-agent case. 3) To solve the corresponding control problem, we propose a novel heterogeneous learning scheme that is particularly tailored to the swarm setting. We evaluate our framework on two well-known system models: the Ising model and the

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

Vicsek model of self-propelled particles. The results demonstrate that our framework is able to produce meaningful reward models from which we can learn local controllers that replicate the observed global system dynamics.

2. THE SWARMDP MODEL

By analogy with the characteristics of natural systems, we characterize a *swarm system* as a collection of agents with the following two properties:

> **Homogeneity:** All agents in a swarm share a common architecture (i.e. they have the same dynamics, degrees of freedom and observation capabilities). As such, they are assumed to be interchangeable.
>
> **Locality:** The agents can observe only parts of the system within a certain range, as determined by their observation capabilities. As a consequence, their decisions depend on their current neighborhood only and not on the whole swarm state.

In principle, any system with these properties can be described as a decentralized partially observable Markov decision process (Dec-POMDP) [21]. However, the homogeneity property, which turns out to be the key ingredient for scalable inference, is not explicitly captured by this model. Since the number of agents contained in a swarm is typically large, it is thus convenient to switch to a more compact system representation that exploits the system symmetries.

For this reason, we introduce a new sub-class of Dec-POMDP models, in the following referred to as *swarMDPs* (Fig. 1), which explicitly implements a homogeneous agent architecture. An agent in this model, which we call a *swarming agent*, is defined as a tuple $\mathbb{A} \coloneqq (\mathcal{S}, \mathcal{O}, \mathcal{A}, R, \pi)$, where:

- $\mathcal{S}, \mathcal{O}, \mathcal{A}$ are sets of local states, observations and actions, respectively.
- $R : \mathcal{O} \to \mathbb{R}$ is an agent-level reward function.
- $\pi : \mathcal{O} \to \mathcal{A}$ is the local policy of the agent which later serves as the decentralized control law of the swarm.

For the sake of simplicity, we consider only reactive policies in this paper, where π is a function of the agent's current observation. Note, however, that the extension to more general policy models (e.g. belief state policies [11] or such that operate on observation histories [21]) is straightforward.

With the definition of the swarming agent at hand, we define a swarMDP as a tuple $(N, \mathbb{A}, T, \xi)$, where:

- N is the number of agents in the system.
- $\mathbb{A}$ is a swarming agent prototype as defined above.
- $T : \mathcal{S}^N \times \mathcal{A}^N \times \mathcal{S}^N \to \mathbb{R}$ is the global transition model of the system. Although T is used only implicitly later on, we can access the conditional probability that the system reaches state $\tilde{s} = (\tilde{s}^{(1)}, \dots, \tilde{s}^{(N)})$ when the agents perform the joint action $a = (a^{(1)}, \dots, a^{(N)})$ at state $s = (s^{(1)}, \dots, s^{(N)})$ as $T(\tilde{s} \mid s, a)$, where $s^{(n)}$, $\tilde{s}^{(n)} \in \mathcal{S}$ and $a^{(n)} \in \mathcal{A}$ represent the local states and the local action of agent n, respectively.
- $\xi : \mathcal{S}^N \to \mathcal{O}^N$ is the observation model of the system.

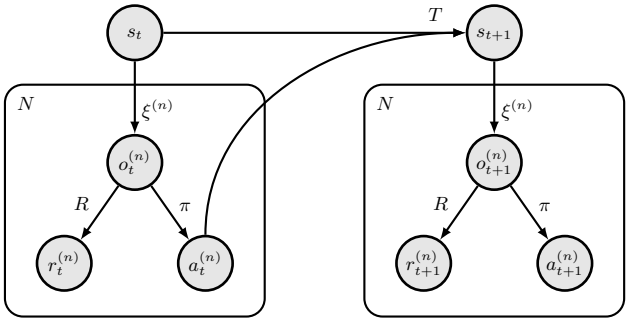


Figure 1: The swarMDP model visualized as a Bayesian network using plate notation. In contrast to a Dec-POMDP, the model explicitly encodes the homogeneity of a swarm.

The observation model ξ tells us which parts of a given system state $s \in \mathcal{S}^N$ can be observed by whom. More precisely, $\xi(s) = (\xi^{(1)}(s), \dots, \xi^{(N)}(s)) \in \mathcal{O}^N$ denotes the ordered set of local observations passed on to the agents at state s. For example, in a school of fish, $\xi^{(n)}$ could be the local alignment of a fish to its immediate neighbors (see Section 4.1). Note that the agents have no access to their local states $s^{(n)} \in \mathcal{S}$ but only to their local observations $o^{(n)} = \xi^{(n)}(s) \in \mathcal{O}$.

It should be mentioned that the observation model can be also defined locally at the agent level, since the observations are agent-related quantities. However, this would still require a global notion of connectivity between the agents, e.g. provided in the form of a dynamic graph which defines the time-varying neighborhood of the agents. Using a global observation model, we can encode all properties in a single object, yielding a more compact system description. Yet, we need to constrain our model class to those models which respect the homogeneity (and thus the interchangeability) of the agents. To be precise, a valid observation model needs to ensure that agent n receives agent m's local observation (and vice versa) if we interchange their local states. Mathematically, this means that any permutation of $s \in \mathcal{S}^N$ must result in the same permutation of $\xi(s)$ – otherwise, the underlying system is not homogeneous. The same property has to hold for the transition model T. A generalization to stochastic observations is possible but not considered in this paper.

3. IRL IN SWARM SYSTEMS

In contrast to existing work on IRL, our goal is *not* to develop a new specialized algorithm that solves the IRL problem in the swarm case. On the contrary, we show that the homogeneity of our model allows us to reduce the multi-agent IRL problem to a single-agent one, for which we can apply a whole class of existing algorithms. This is possible since, at its heart, the underlying control problem of the swarMDP is intrinsically a single-agent problem because all agents share the same policy.[1] In the subsequent sections, we show that this symmetry property also translates to the value functions of the agents. Algorithmically, we exploit the fact that most existing IRL methods, such as [2, 19, 20, 24, 29, 35], share a common generic form (Algorithm 1), which involves three main steps [17]: 1) policy update 2) value estimation and 3) reward update. The important detail to

[1] However, the decentralized nature of the problem remains!

note is that only the first two steps of this procedure are system-specific while the third step is, in fact, independent of the target system (see references listed above for details). Consequently, our problem reduces to finding swarm-based solutions for the first two steps such that the overall procedure returns a "meaningful" reward model in the IRL context. The following sections discuss these steps in detail.

Algorithm 1: GENERIC IRL

Input: expert data $\mathcal{D}$, MDP without reward function

0: Initialize reward function $R^{\{0\}}$
   for $i = 0, 1, 2, ...$
1: Policy update: Find optimal policy $\pi^{\{i\}}$ for $R^{\{i\}}$
2: Value estimation: Compute corresponding value $V^{\{i\}}$
3: Reward update: Given $V^{\{i\}}$ and $\mathcal{D}$, compute $R^{\{i+1\}}$

3.1 Policy Update

We start with the policy update step, where we are faced with the problem of learning a suitable system policy for a given reward function. For this purpose, we first need to define a suitable learning objective for the swarm setting in the context of the IRL procedure. In the next paragraphs, we show that the homogeneity property of our model naturally induces such a learning objective, and we furthermore present a simple learning strategy to optimize this objective.

3.1.1 Private Value & Bellman Optimality

Analogous to the single-agent case [28], we define the *private value* of an agent n at a swarm state $s \in \mathcal{S}^N$ under policy π as the expected sum of discounted rewards accumulated by the agent over time, given that all agents execute π,

$$V^{(n)}(s \mid \pi) \coloneqq \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R(\xi^{(n)}(s_{t+k})) \mid \pi, s_t = s\right], \quad (1)$$

Herein, $\gamma \in [0, 1)$ is a discount factor, and the expectation is with respect to the random system trajectory starting from s. Note that, due to the assumed time-homogeneity of the transition model T, the above definition of value is, in fact, independent of any particular starting time t. Denoting further by $Q^{(n)}(s, a \mid \pi)$ the state-action value of agent n at state s for the case that all agents execute policy π, except for agent n who performs action $a \in \mathcal{A}$ once and follows π thereafter, we obtain the following Bellman equations:

$$V^{(n)}(s \mid \pi) = R(\xi^{(n)}(s)) + \gamma \sum_{\tilde{s} \in \mathcal{S}^N} P(\tilde{s} \mid s, \pi) V^{(n)}(\tilde{s} \mid \pi),$$

$$Q^{(n)}(s, a \mid \pi) = R(\xi^{(n)}(s)) + \gamma \sum_{\tilde{s} \in \mathcal{S}^N} P^{(n)}(\tilde{s} \mid s, a, \pi) V^{(n)}(\tilde{s} \mid \pi).$$

Here, $P(\tilde{s} \mid s, \pi)$ denotes the probability of reaching swarm state $\tilde{s}$ from s when *every* agent performs policy π and, analogously, $P^{(n)}(\tilde{s} \mid s, a, \pi)$ denotes the probability of reaching swarm state $\tilde{s}$ from state s if agent n chooses action a and *all other* agents execute policy π. Note that both these objects are implicitly defined via the transition model T.

3.1.2 Local Value

Unfortunately, the value function in Eq. (1) is not locally plannable by the agents since they have no access to the global swarm state s. From a control perspective, we thus require an alternative notion of optimality that is based on local information only and, hence, computable by the agents. Analogous to the belief value in single-agent systems [14, 15], we therefore introduce the following *local value function*,

$$V_t^{(n)}(o \mid \pi) \coloneqq \mathbb{E}_{P_t(s \mid o^{(n)} = o, \pi)}\left[V^{(n)}(s \mid \pi)\right],$$

which represents the expected return of agent n under consideration of its current local observation of the global system state. In our next proposition, we highlight two key properties of this quantity: 1) It is not only locally plannable but also reduces the multi-agent problem to a single-agent one in the sense that all local values coincide. 2) In contrast to the private value, the local value is time-dependent because the conditional probabilities $P_t(s \mid o^{(n)} = o, \pi)$, in general, depend on time. However, it converges to a stationary value asymptotically under suitable conditions.

Proposition 1. *Consider a swarMDP as defined above and the stochastic process $\{S_t\}_{t=0}^{\infty}$ of the swarm state induced by the system policy π. If the initial state distribution of the system is invariant under permutation[2] of agents, all local value functions are identical,*

$$V_t^{(m)}(o \mid \pi) = V_t^{(n)}(o \mid \pi) \quad \forall m, n. \quad (2)$$

In this case, we may drop the agent index and denote the common local value function as $V_t(o \mid \pi)$. If, furthermore, it holds that $S_t \xrightarrow{a.s.} S$ for some S with law P and the common local value function is continuous almost everywhere (i.e. its set of discontinuity points is P-null) and bounded above, then the local value function $V_t(o \mid \pi)$ will converge to a limit,

$$V_t(o \mid \pi) \to V(o \mid \pi), \quad (3)$$

where $V(o \mid \pi) = \mathbb{E}_{P(s \mid o^{(n)} = o, \pi)}\left[V^{(n)}(s \mid \pi)\right]$.

Proof. Fix any two agents, say agent 1 and 2. For these agents, define a permutation operation $\sigma : \mathcal{S}^N \to \mathcal{S}^N$ as

$$\sigma(s) \coloneqq (s^{(2)}, s^{(1)}, s^{(3)}, \ldots, s^{(N)}),$$

where $s = (s^{(1)}, s^{(2)}, s^{(3)}, \ldots, s^{(N)})$. Due to the homogeneity of the system, i.e. since $R(\xi^{(1)}(s)) = R(\xi^{(2)}(\sigma(s)))$ and $P(\tilde{s} \mid s, \pi) = P(\sigma(\tilde{s}) \mid \sigma(s), \pi)$, it follows immediately that $V^{(1)}(s \mid \pi) = V^{(2)}(\sigma(s) \mid \pi)\ \forall s$. This essentially means: the value assigned to agent 1 at swarm state s is the same as the value that would be assigned to agent 2 if we interchanged their local states, i.e. at state $\sigma(s)$. Note that this is effectively the same as renaming the agents. The homogeneity of the system ensures that the symmetry of the initial state distribution $P_0(s)$ is maintained at all subsequent points in time, i.e. $P_t(s \mid \pi) = P_t(\sigma(s) \mid \pi)\ \forall s, t$. In particular, it holds that $P_t(s \mid o^{(1)} = o, \pi) = P_t(\sigma(s) \mid o^{(2)} = o, \pi)\ \forall s, t$ and, accordingly,

$$\begin{aligned}
&V_t^{(1)}(o \mid \pi) - V_t^{(2)}(o \mid \pi) \\
&= \mathbb{E}_{P_t(s \mid o^{(1)} = o, \pi)}\left[V^{(1)}(s \mid \pi)\right] - \mathbb{E}_{P_t(s \mid o^{(2)} = o, \pi)}\left[V^{(2)}(s \mid \pi)\right] \\
&= \sum_{s \in \mathcal{S}^N} \Big(P_t(s \mid o^{(1)} = o, \pi) V^{(1)}(s \mid \pi) \ \ldots \\
&\qquad \ldots \ - \ P_t(\sigma(s) \mid o^{(2)} = o, \pi) V^{(2)}(\sigma(s) \mid \pi)\Big) = 0,
\end{aligned}$$

[2] Since we assume that the agents are interchangeable, it follows naturally to consider only permutation-invariant initial distributions.

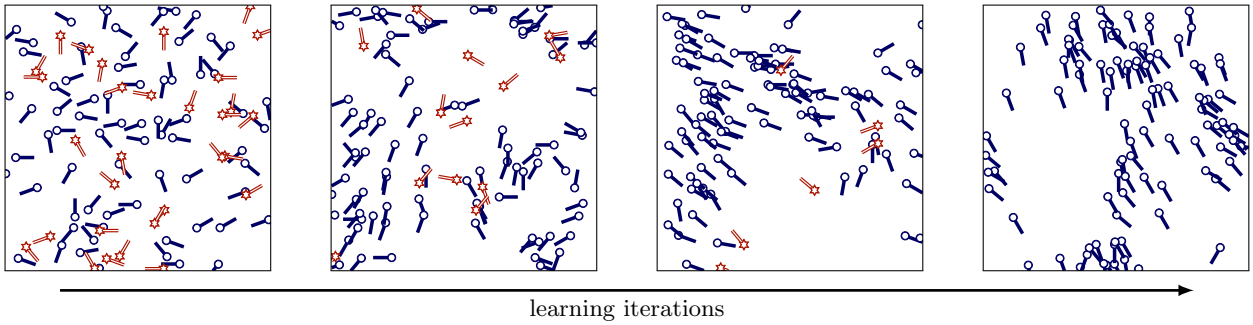


Figure 2: Snapshots of the proposed learning scheme applied to the Vicsek model (Section 4.1). The agents are divided into a greedy set ( ) and an exploration set ( ) to facilitate the exploration of locally desynchronized swarm states. The size of the exploration set is reduced over time to gradually transfer the system into a homogeneous stationary behavior.

which shows that the local value functions are identical for all agents. Treating the value as a random variable and using the fact that it is continuous almost everywhere, it follows that $V^{(n)}(S_t \mid \pi) \xrightarrow{a.s.} V^{(n)}(S \mid \pi)$ since $S_t \xrightarrow{a.s.} S$. As we assume the function to be finite, i.e. $|V^{(n)}(S_t \mid \pi)| < V^*$ for some $V^* \in \mathbb{R}$, it holds by conditional dominated convergence theorem [4] that $\mathbb{E}\left[V^{(n)}(S_t \mid \pi) \mid o^{(n)} = o, \pi\right] \to \mathbb{E}\left[V^{(n)}(S \mid \pi) \mid o^{(n)} = o, \pi\right]$, i.e. $V_t(o \mid \pi) \to V(o \mid \pi)$. □

3.1.3 Heterogeneous Q-learning

With the local value in Eq. (2), we have introduced a system-wide performance measure which can be evaluated at the agent level and, hence, can be used by the agents for local planning. Yet, its computation involves the evaluation of an expectation with respect to the current swarm state of the system. This requires the agents to maintain a belief about the global system state at any point in time to coordinate their actions, which itself is a hard problem.[3] However, for many swarm-related tasks (e.g. consensus problems [26]), it is sufficient to optimize the stationary behavior of the system. This is, in fact, a much easier task since it allows us to forget about the temporal aspect of the problem.

In this section, we present a comparably simple learning method, specifically tailored to the swarm setting, which solves this task by optimizing the system's stationary value in Eq. (3). Similar to the local value function, we start by defining a *local Q-function* for each agent,

$$Q_t^{(n)}(o, a \mid \pi) := \mathbb{E}_{P_t(s \mid o^{(n)} = o, \pi)}\left[Q^{(n)}(s, a \mid \pi)\right],$$

which assesses the quality of a particular action played by agent n at time t. Following the same line of argument as before, one can show that these Q-functions are again identical for all agents and, moreover, that they converge to the following asymptotic value function,

$$Q(o, a \mid \pi) = \mathbb{E}_{P(s \mid o^{(n)} = o, \pi)}\left[Q^{(n)}(s, a \mid \pi)\right], \qquad (4)$$

which can be understood as the state-action value of a generic agent that is coupled to a stationary field generated by and executing policy π. In the following, we pose the task of optimizing this Q-function as a game-theoretic one. To be precise, we consider a hypothetical game between each agent and the environment surrounding it, where the agent plays the optimal response to this stationary field,

$$\pi_{\mathrm{R}}(o \mid \pi) := \arg\max_{a \in \mathcal{A}} Q(o, a \mid \pi),$$

and the environment reacts with a new swarm behavior generated by this policy. By definition, any optimal system policy π^* describes a fixed-point of this game,

$$\pi_{\mathrm{R}}(o \mid \pi^*) = \pi^*(o),$$

which motivates the following iterative learning scheme: Starting with an arbitrary initial policy, we run the system until it reaches its stationary behavior and estimate the corresponding asymptotic Q-function. Based on this Q-function, we update the system policy according to the best response operator defined above. The updated policy, in turn, induces a new swarm behavior for which we estimate a new Q-function, and so on. As soon as we reach a fixed-point, the system has arrived at an optimal behavior in the form of a symmetric Nash equilibrium where all agents collectively execute a policy which, for each agent individually, provides the optimal response to the other agents' behavior.

However, the following practical problems remain: 1) In general, it can be time-consuming to wait for the system to reach its stationary behavior at each iteration of the algorithm. 2) At stationarity, we need a way to estimate the corresponding stationary Q-function. Note that this involves both estimating the Q-values of actions that are dictated by the current policy as well as Q-values of actions that deviate from the current behavior, which requires a certain amount of exploratory moves. As a solution to both problems, we propose the following *heterogeneous learning scheme*, which artificially breaks the symmetry of the system by separating the agents into two disjoint groups: a greedy set and an exploration set. While the agents in the greedy set provide a reference behavior in the form of the optimal response to the current Q-function shared between all agents, the agents in the exploration set randomly explore the quality of different actions in the context of the current system policy. At each iteration, the gathered experience of all agents is processed sequentially via the following Q-update [32],

$$\hat{Q}(o_t^{(n)}, a_t^{(n)}) \leftarrow (1-\alpha)\hat{Q}(o_t^{(n)}, a_t^{(n)}) + \alpha\big(r_t^{(n)} + \gamma \max_{a \in \mathcal{A}} \hat{Q}(o_{t+1}^{(n)}, a)\big),$$

with learning rate $\alpha \in (0, 1)$. Over time, more and more

[3] In principle, this is possible since – in contrast to a Dec-POMDP – each agent knows the policy of the other agents.

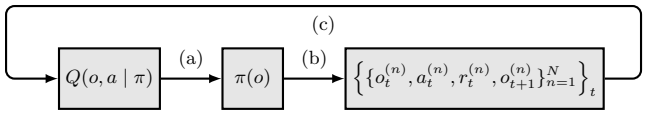


Figure 3: Pictorial description of the proposed learning scheme. (a) The next policy is obtained from the current estimate of the system's stationary Q-function. (b) Heterogeneous one-step transition of the system. (c) The estimate of the Q-function is updated based on the new experience.

Algorithm 2: HETEROGENEOUS Q-LEARNING

Input: swarMDP without policy π

0: Initialize shared Q-function, learning rate and fraction of exploring agents (called the temperature)
for $i = 0, 1, 2, ...$
1: Separate the swarm into exploring and greedy agents according to the current temperature
2: Based on the current swarm state and Q-function, select actions for all agents
3: Iterate the system and collect rewards
4: Update the Q-function based on the new experience
5: Decrease the learning rate and the temperature

exploring agents are assigned to the greedy set so that the system is gradually transferred into a *homogeneous stationary* regime and thereby smoothly guided towards a fixed-point policy (see Figure 2). Herein, the learning rate α naturally reduces the influence of experience acquired at early (non-synchronized) stages of the system, which allows us to update the system policy without having to wait until the swarm converges to its stationary behavior.

The heterogeneity of the system during the learning phase ensures that also locally desynchronized swarm states are well-explored (together with their local Q-values) so that the agents can learn adequate responses to out-of-equilibrium situations. This phenomenon is best illustrated by the agent constellation in third sub-figure of Figure 2. It shows a situation that is highly unlikely under a homogeneous learning scheme as it requires a series of consecutive exploration steps by only a few agents while all their neighbors need to behave consistently optimally at the same time. The final procedure, which can be interpreted as a model-free variant of policy iteration [12] in a non-stationary environment, is summarized in Algorithm 2, together with a pictorial description of the main steps in Fig. 3. While we cannot provide a convergence proof at this stage, the algorithm converged in all our simulations and generated policies with a performance close to that of the expert system (see Section 4).

3.2 Value Estimation

In the last section, we have shown a way to implement the policy update in Algorithm 1 based on local information acquired at the agent level. Next, we need to assign a suitable value to the obtained policy which allows a comparison to the expert behavior in the subsequent reward update step.

3.2.1 Global Value

The comparison of the learned behavior and the expert behavior should take place on a global level, since we want the updated reward function to cause a new system behavior which mimics the expert behavior *globally*. Therefore, we introduce the following *global value*,

$$V_{|\pi}^{(n)} \coloneqq \mathbb{E}_{P_0(s)}\left[V^{(n)}(s \mid \pi)\right],$$

which represents the expected return of an agent under π, averaged over all possible initial states of the swarm. From the system symmetry, i.e. since $P_0(s) = P_0(\sigma(s))$, it follows immediately that this global value is independent of the specific agent under consideration,

$$V_{|\pi}^{(m)} = V_{|\pi}^{(n)} \quad \forall m, n.$$

Hence, the global value should be considered as a system-related performance measure (as opposed to an agent-specific property), which may be utilized for the reward update in the last step of the algorithm. We can construct an unbiased estimator for this quantity from any local agent trajectory,

$$\hat{V}_{|\pi}^{(n)} = \sum_{t=0}^{\infty} \gamma^t R(\xi^{(n)}(s_t)) = \sum_{t=0}^{\infty} \gamma^t r_t^{(n)}. \tag{5}$$

Since all local estimators are identically distributed, we can increase the accuracy of our estimate by considering the information provided by the whole swarm,

$$\hat{V}_{|\pi} = \frac{1}{N}\sum_{n=1}^{N} \hat{V}_{|\pi}^{(n)} = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{\infty} \gamma^t r_t^{(n)}. \tag{6}$$

Note, however, that the local estimators are not independent since all agents are correlated through the system process. Nevertheless, due to the local coupling structure of a swarm, this correlation is caused only *locally*, which means that the correlation between any two agents will drop when their topological distance increases. We demonstrate this phenomenon for the Vicsek model in Section 4.1.

3.3 Reward Update

The last step of Algorithm 1 consists in updating the estimated reward function. Depending on the single-agent IRL framework in use, this involves an algorithm-specific optimization procedure, e.g. in the form of a quadratic program [2, 20] or a gradient-based optimization [19, 35]. For our experiments in Section 4, we follow the max-margin approach presented in [2]; however, the procedure can be replaced with other value-based methods (see Section 3).

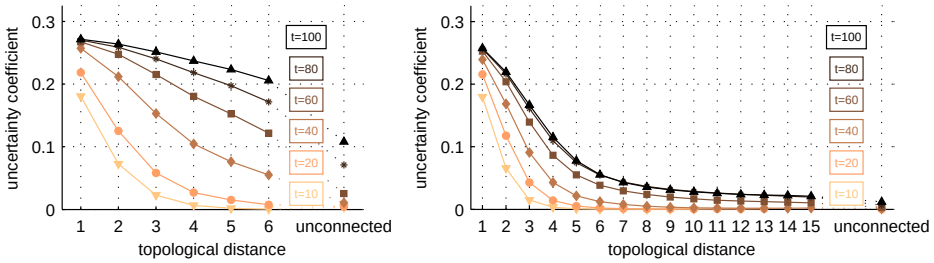
For this purpose, the local reward function is represented as a linear combination of observational features, $R(o) = w^\top \phi(o)$, with weights $w \in \mathbb{R}^d$ and a given feature function $\phi : \mathcal{O} \to \mathbb{R}^d$. The feature weights after the ith iteration of Algorithm 1 are then obtained as

$$w^{\{i+1\}} = \underset{w:||w||_2 \leq 1}{\arg\max} \min_{j \in \{1,\dots,i\}} w^\top(\mu_E - \mu^{(j)}).$$

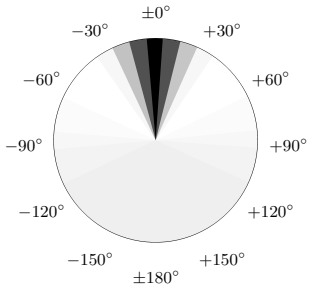
where μ_E and $\{\mu^{(j)}\}_{j=1}^i$ are the *feature expectations* [2] of the expert policy and the learned policies up to iteration i. Simulating a one-shot learning experiment, we estimate these quantities from a single system trajectory based on Eq. (6),

$$\hat{\mu}(\pi) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{\infty} \gamma^t \phi(\xi^{(n)}(s_t)).$$

where the state sequence $(s_0, s_1, s_2, \dots)$ is generated using the respective policy π. For more details, we refer to [2].



(a) Uncertainty coefficient for the interaction radii $\rho = 0.125$ (left) and $\rho = 0.05$ (right). The values are based on a kernel density estimate of the joint distribution of two agents' headings.



(b) Learned reward model.

Figure 4: Simulation results for the Vicsek model. (a) The system's uncertainty coefficient as a function of the topological distance between two agents, estimated from 10000 Monte Carlo runs of the expert system. (b) Learned reward model as a function of an agent's local misalignment, averaged over 100 Monte Carlo experiments. Dark color indicates high reward.

4. SIMULATION RESULTS

In this section, we provide simulation results for two different system types. For the policy update, the initial number of exploring agents is set to 50% of the population size and the learning rate is initialized close to 1. Both quantities are controlled by a quadratic decay which ensures that, at the end of the learning period, i.e. after 200 iterations, the learning rate reaches zero and there are no exploring agents left. Note that these parameters are by no means optimized; yet, in our experiments we observed that the learning results are largely insensitive to the particular choice of values. Since the agents' observation space is one-dimensional in both experiments, we use a simple tabular representation for the learned Q-function; for higher-dimensional problems, one needs to resort to function approximation [12]. Videos can be found at `http://www.spg.tu-darmstadt.de/aamas2017`.

4.1 The Vicsek Model

First, we test our framework on the Vicsek model of self-propelled particles [30]. The model consists of a fixed number of particles, or agents, living in the unit square $[0, 1] \times [0, 1]$ 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, as summarized by the local state variable $s_t^{(n)} := (x_t^{(n)}, \theta_t^{(n)})$. The time-varying neighborhood structure of the agents is determined by a fixed 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 $\{\Delta\theta_t^{(n)}\}$,

$$\begin{aligned} \theta_{t+1}^{(n)} &= \langle \theta_t^{(n)} \rangle_\rho + \Delta\theta_t^{(n)}, \\ x_{t+1}^{(n)} &= x_t^{(n)} + v_t^{(n)}. \end{aligned} \tag{7}$$

Herein, $\langle \theta_t^{(n)} \rangle_\rho$ denotes the mean orientation of all agents within the ρ-neighborhood of agent n at time t, and $v_t^{(n)} = v \cdot [\cos\theta_t^{(n)}, \sin\theta_t^{(n)}]$ is the velocity vector of agent n.

Our goal is to learn a model for this expert behavior from recorded agent trajectories using the proposed framework. As a simple observation mechanism, we let the agents in our model compute the angular distance to the average orientation of their neighbors, i.e. $o_t^{(n)} = \xi^{(n)}(s_t) := \langle \theta_t^{(n)} \rangle_\rho - \theta_t^{(n)}$, giving them the ability to monitor their local misalignment. For simplicity, we discretize the observation space $[0, 2\pi)$ into 36 equally-sized intervals (Fig. 6), corresponding to the features ϕ (Section 3.3). Furthermore, we coarse-grain the space of possible direction changes to $[-60°, -50°, \ldots, 60°]$, resulting in a total of 13 actions available to the agents. For the experiment, we use a system size of $N = 200$, an interaction radius of $\rho = 0.1$ (if not stated otherwise), an absolute velocity of $v = 0.1$, a discount factor of $\gamma = 0.9$, and a zero-mean Gaussian noise model for $\{\Delta\theta_t^{(n)}\}$ with a standard deviation of 10°. These parameter values are chosen such that the expert system operates in an ordered phase [30].

Local Coupling & Redundancy

In Section 3.2.1 we claimed that, due to the local coupling in a swarm, the correlation between any two agents will decrease with growing topological distance. In this section, we substantiate our claim by analyzing the coupling strength in the system as a function of the topological distance between the agents. As a measure of (in-)dependence, we employ the *uncertainty coefficient* [23], a normalized version of the mutual information, which reflects the amount of information we can predict about an agent's orientation by observing that of another agent. As opposed to linear correlation, this measure is able to capture non-linear dependencies and is, hence, more meaningful in the context of the Vicsek model whose state dynamics are inherently non-linear.

Figure 4(a) depicts the result of our analysis which nicely reveals the spatio-temporal flow of information in the system. It confirms that the mutual information exchange between the agents strongly depends on the strength of their coupling which is determined by 1) their topological distance and 2) the number of connecting links (seen from the fact that, for a fixed distance, the dependence grows with the interaction radius). We also see that, for increasing radii, the dependence grows even for agents that are temporarily not connected through the system, due to the increasing chances of having been connected at some earlier stage.

Learning Results

An inherent problem with any IRL approach is the assessment of the extracted reward function as there is typically no ground truth to compare with. The simplest way to check the plausibility of the result is by subjective inspection: since a system's reward function can be regarded as a concise description of the task being performed, the estimate should

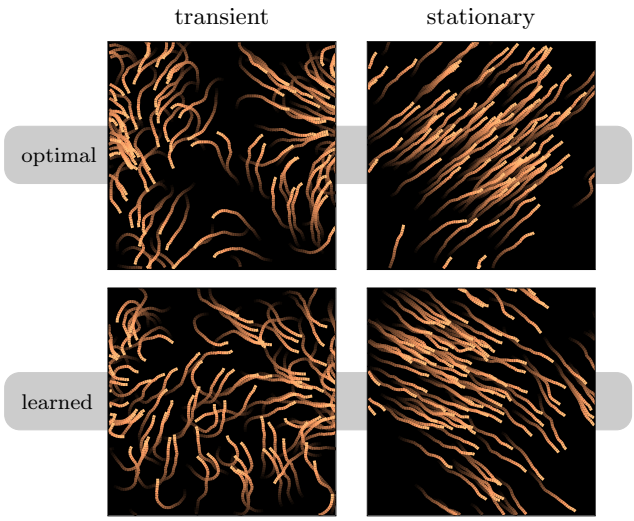


Figure 5: Illustrative trajectories of the Vicsek model generated under the optimal policy and a learned policy. A color-coding scheme is used to indicate the temporal progress.

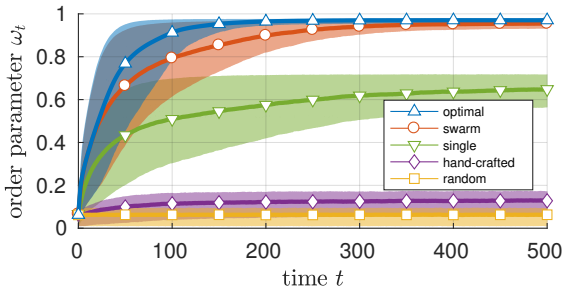


Figure 6: Slopes of the order parameter ω_t in the Vicsek model. From top to bottom, the curves show the results for the expert policy, the learned IRL policy, the result we get when the feature expectations are estimated from just one single agent, for a hand-crafted reward function, and for random policies. For the optimal policy, we show the empirical mean and the corresponding 10% and 90% quantiles, based on 10000 Monte Carlo runs. For the learned policies, we instead show the average over 100 conditional quantiles (since the outcome of the learning process is random), each based on 100 Monte Carlo runs with a fixed policy.

explain the observed system behavior reasonably well. As we can see from Figure 4(b), this is indeed the case for the obtained result. Although there is no "true" reward model for the Vicsek system, we can see from the system equations in (7) that the agents tend to align over time. Clearly, such dynamics can be induced by giving higher rewards for synchronized states and lower (or negative) rewards for misalignment. Inspecting further the induced system dynamics (Fig. 5), we observe that the algorithm is able to reproduce the behavior of the expert system, both during the transient phase and at stationarity. Note that the absolute direction of travel is not important here as the model considers only relative angles between the agents. Finally, we compare the results in terms of the *order parameter* [30], which provides a measure for the total alignment of the swarm,

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

with values close to 1 indicating strong synchronization of the system. Figure 6 depicts its slope for different system policies, including the expert policy and the learned ones. From the result, we can see a considerable performance gain for the proposed value estimation scheme (Eq. (6)) as compared to a single-agent approach (Eq. (5)). This again confirms our findings from the previous section since the increase in performance has to stem from the additional information provided by the other agents. As a further reference, we also show the result for a hand-crafted reward model, where we provide a positive reward only if the local observation of an agent falls in the discretization interval centered around 0° misalignment. As we can see, the learned reward model significantly outperforms the ad-hoc solution.

4.2 The Ising Model

In our second experiment, we apply the IRL framework to the well-known Ising model [10] which, in our case, consists of a finite grid of atoms (i.e. agents) of size 100×100. Each agent has an individual spin $q_t^{(n)} \in \{+1, -1\}$ which, together with its position on the grid, forms its local state, $s_t^{(n)} := (x^{(n)}, y^{(n)}, q_t^{(n)})$. For our experiment, we consider a static 5×5-neighborhood system, meaning that each agent interacts only with its 24 closest neighbors (i.e. agents with a maximum Chebyshev distance of 2). Based on this neighborhood structure, we define the global system energy as

$$E_t := \sum_{n=1}^{N} \sum_{m \in \mathcal{N}_n} \mathbb{1}(q_t^{(n)} \neq q_t^{(m)}) = \sum_{n=1}^{N} E_t^{(n)},$$

where $\mathcal{N}_n$ and $E_t^{(n)}$ are the neighborhood and local energy contribution of agent n, and $\mathbb{1}(\cdot)$ denotes the indicator function. Like the order parameter for the Vicsek model, the global energy serves as a measure for the total alignment of the system, with zero energy indicating complete state synchronization. In our experiment, we consider two possible actions available to the agents, i.e. *keep the current spin* and *flip the spin*. The system dynamics are chosen such that the agent transitions to the desired state with probability 1. As before, we give the agents the ability to monitor their local misalignment, this time provided in the form of their individual energy contributions, i.e. $o_t^{(n)} = \xi^{(n)}(s_t) := E_t^{(n)}$.

A meaningful goal for the system is to reach a global state configuration of minimum energy. Again, we are interested in learning a behavioral model for this task from expert trajectories. In this case, our expert system performs a local majority voting using a policy which lets the agents adopt the spin of the majority of their neighbors. Essentially, this policy implements a synchronous version of the *iterated conditional modes* algorithm [3], which is guaranteed to translate the system to a state of locally minimum energy.

Figures 7 and 8 depict, respectively, the learned mean reward function and the slopes of the global energy for the different policies. As in the previous example, the extracted reward function explains the expert behavior well[4] and we

[4]Note that assigning a neutral reward to states of high local energy is reasonable, since a strong local misalignment indicates high synchronization of the opposite spin in the neighborhood.

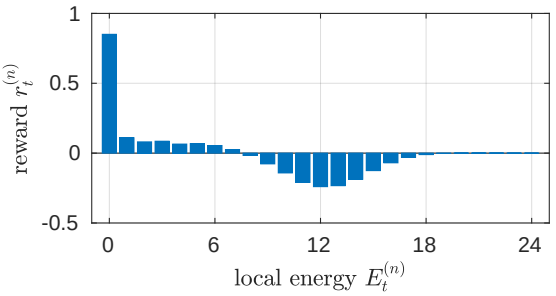


Figure 7: Learned reward function for the Ising model, averaged over 100 Monte Carlo experiments.

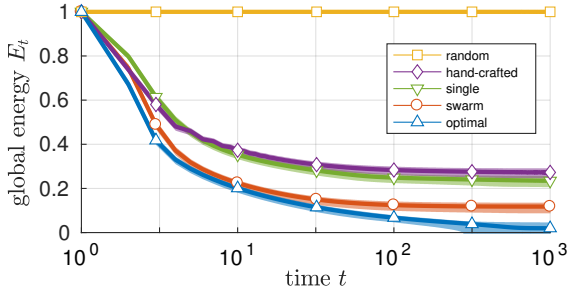


Figure 8: Slopes of the global energy E_t in the Ising model. The graphs are analogous to those in Figure 6.

observe the same qualitative performance improvement as for the Vicsek system, both when compared to the single-agent estimation scheme and to the hand-crafted model.

5. CONCLUSION & DISCUSSION

Our objective in this paper has been to extend the concept of IRL to homogeneous multi-agent systems, called swarms, in order to learn a local reward function from observed global dynamics that is able to explain the emergent behavior of a system. By exploiting the homogeneity of the newly introduced swarMDP model, we showed that both value estimation and policy update required for the IRL procedure can be performed based on local experience gathered at the agent level. The so-obtained reward function was provided as input to a novel learning scheme to build a local policy model which mimics the expert behavior. We demonstrated our framework on two types of system dynamics where we achieved a performance close to that of the expert system.

Nevertheless, there remain some open questions. In the process of IRL, we have tacitly assumed that the expert behavior can be reconstructed based on local interactions. Of course, this is a reasonable assumption for self-organizing systems which naturally operate in a decentralized manner. For arbitrary expert systems, however, we cannot exclude the possibility that the agents are instructed by a central controller which has access to the global system state. This brings us back to the following questions: When is it possible to reconstruct global behavior based on local information? If it is not possible for a given task, how well can we approximate the centralized solution by optimizing local values?

In an attempt to understand the above mentioned questions, we propose the following characterization of the reward function that would make a local policy optimal in a swarm. To this end, we enumerate the swarm states and observations by $\mathcal{S}^N = \{s_i\}_{i=1}^K$ and $\mathcal{O} = \{o_i\}_{i=1}^L$, respectively. Furthermore, we fix an agent n and define matrices $\mathbf{P}_o$ and $\{\mathbf{P}_a\}_{a=1}^{|\mathcal{A}|}$, where $[\mathbf{P}_o]_{ij} = P(s_j \mid o^{(n)} = o_i, \pi)$ and $[\mathbf{P}_a]_{ij} = P^{(n)}(s_j \mid s_i, a, \pi)$. Finally, we represent the reward function as a vector, i.e. $\mathbf{R} = (R(\xi^{(n)}(s_1)), \dots, R(\xi^{(n)}(s_K)))^\mathsf{T}$.

Proposition 2. *Consider a swarm $(N, \mathbb{A}, T, \xi)$ of agents $\mathbb{A} = (\mathcal{S}, \mathcal{O}, \mathcal{A}, R, \pi)$ and a discount factor $\gamma \in [0, 1)$. Then, a policy $\pi : \mathcal{O} \to \mathcal{A}$ given by $\pi(o) := a_1$ is optimal[5] with respect to $V(o \mid \pi)$ if and only if the reward R satisfies*

$$\mathbf{P}_o(\mathbf{P}_{a_1} - \mathbf{P}_a)(\mathbf{I} - \gamma\mathbf{P}_{a_1})^{-1}\mathbf{R} \geq 0 \quad \forall a \in \mathcal{A}. \tag{8}$$

[5]We can ensure that $\pi(o) = a_1$ by renaming actions accordingly [20].

Proof. Expressing Eq. (1) using vector notation, we get

$$\mathbf{V}_{s|\pi} = (\mathbf{I} - \gamma\mathbf{P}_{a_1})^{-1}\mathbf{R},$$

where $\mathbf{V}_{s|\pi} = (V^{(n)}(s_1 \mid \pi), \dots, V^{(n)}(s_K \mid \pi))^\mathsf{T}$. According to Prop. 1, the corresponding limiting value function is

$$V(o \mid \pi) = \sum_{i=1}^K P(s_i \mid o^{(n)} = o, \pi) V^{(n)}(s_i \mid \pi).$$

Rewritten in vector notation, we obtain

$$\mathbf{V}_{o|\pi} = \mathbf{P}_o \mathbf{V}_{s|\pi}, \tag{9}$$

where $\mathbf{V}_{o|\pi} = (V(o_1 \mid \pi), \dots, V(o_L \mid \pi))^\mathsf{T}$. Now, $\pi(o) = a_1$ is optimal if and only if for all $a \in \mathcal{A}, o \in \mathcal{O}$

$$\begin{aligned}
&Q^{(n)}(o, a_1 \mid \pi) \geq Q^{(n)}(o, a \mid \pi) \\
&\Leftrightarrow \sum_{i=1}^K P(s_i \mid o^{(n)} = o, \pi) \sum_{j=1}^K P^{(n)}(s_j \mid s_i, a_1, \pi) V^{(n)}(s_j \mid \pi) \\
&\quad \geq \sum_{i=1}^K P(s_i \mid o^{(n)} = o, \pi) \sum_{j=1}^K P^{(n)}(s_j \mid s_i, a, \pi) V^{(n)}(s_j \mid \pi) \\
&\Leftrightarrow \mathbf{P}_o(\mathbf{P}_{a_1} - \mathbf{P}_a)\mathbf{V}_{s|\pi} \geq 0 \\
&\Leftrightarrow \mathbf{P}_o(\mathbf{P}_{a_1} - \mathbf{P}_a)(\mathbf{I} - \gamma\mathbf{P}_{a_1})^{-1}\mathbf{R} \geq 0.
\end{aligned}$$

□

Remark. *Following a similar derivation as in [20], we obtain the characterization set with respect to $\mathbf{V}_{s|\pi}$ as*

$$(\mathbf{P}_{a_1} - \mathbf{P}_a)(\mathbf{I} - \gamma\mathbf{P}_{a_1})\mathbf{R} \geq 0. \tag{10}$$

Notice that, as Eq. (10) implies Eq. (8), an $\mathbf{R}$ that makes $\pi(o)$ optimal for $\mathbf{V}_{s|\pi}$, also makes it optimal for $\mathbf{V}_{o|\pi}$. Therefore, denoting by $\mathcal{R}_L$ and $\mathcal{R}_G$ the solution sets corresponding to the local and global values $\mathbf{V}_{o|\pi}$ and $\mathbf{V}_{s|\pi}$, we conclude

$$\mathcal{R}_G \subseteq \mathcal{R}_L,$$

with equality in the trivial case where observation o is sufficient to determine the swarm state s. It is therefore immediate that, as long as there is uncertainty about the swarm state, local planning can only guarantee globally optimal behavior in an average sense as pronounced by $\mathbf{P}_o$ (see Eq. (9)).

Acknowledgment

W. R. KhudaBukhsh was supported by the German Research Foundation (DFG) within the Collaborative Research Center (CRC) 1053 – MAKI. H. Koeppl acknowledges the support of the LOEWE Research Priority Program CompuGene.

REFERENCES

[1] P. Abbeel, A. Coates, and A. Y. Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 2010.

[2] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proc. 21st International Conference on Machine Learning*, page 1, 2004.

[3] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 259–302, 1986.

[4] P. Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2013.

[5] J. Buhl, D. Sumpter, I. D. Couzin, J. J. Hale, E. Despland, E. Miller, and S. J. Simpson. From disorder to order in marching locusts. *Science*, 312(5778):1402–1406, 2006.

[6] I. D. Couzin. Collective cognition in animal groups. *Trends in cognitive sciences*, 13(1):36–43, 2009.

[7] L. Dufton and K. Larson. Multiagent policy teaching. In *Proc. 8th International Conference on Autonomous Agents and Multiagent Systems*, 2009.

[8] R. A. Freitas. Current status of nanomedicine and medical nanorobotics. *Journal of Computational and Theoretical Nanoscience*, 2(1):1–25, 2005.

[9] S. C. Goldstein, J. D. Campbell, and T. C. Mowry. Programmable matter. *Computer*, 38(6):99–101, 2005.

[10] E. Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, 1925.

[11] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

[12] M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4(Dec):1107–1149, 2003.

[13] V. Lesser, C. L. Ortiz J., and M. Tambe. *Distributed sensor networks: a multiagent perspective*, volume 9. Springer Science & Business Media, 2012.

[14] F. S. Melo. Exploiting locality of interactions using a policy-gradient approach in multiagent learning. In *Proc. 18th European Conference on Artificial Intelligence*, page 157, 2008.

[15] N. Meuleau, L. Peshkin, K.-E. Kim, and L. P. Kaelbling. Learning finite-state controllers for partially observable environments. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence*, pages 427–436, 1999.

[16] D. Michie, M. Bain, and J. Hayes-Miches. Cognitive models from subcognitive skills. *IEE Control Engineering Series*, 44:71–99, 1990.

[17] B. Michini and J. P. How. Bayesian nonparametric inverse reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer, 2012.

[18] S. Natarajan, G. Kunapuli, K. Judah, P. Tadepalli, K. Kersting, and J. Shavlik. Multi-agent inverse reinforcement learning. In *Proc. 9th International Conference on Machine Learning and Applications*, pages 395–400, 2010.

[19] G. Neu and C. Szepesvari. Apprenticeship learning using inverse reinforcement learning and gradient methods. In *Proc. 23rd Conference on Uncertainty in Artificial Intelligence*, pages 295–302, 2007.

[20] A. Y. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *Proc. 17th International Conference on Machine Learning*, pages 663–670, 2000.

[21] F. A. Oliehoek. *Decentralized POMDPs*, pages 471–503. Springer, 2012.

[22] E. Omel'chenko, Y. L. Maistrenko, and P. A. Tass. Chimera states: the natural link between coherence and incoherence. *Physical review letters*, 100(4):044105, 2008.

[23] W. H. Press. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge University Press, 2007.

[24] D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In *Proc. 20th International Joint Conference on Artifical Intelligence*, pages 2586–2591, 2007.

[25] T. S. Reddy, V. Gopikrishna, G. Zaruba, and M. Huber. Inverse reinforcement learning for decentralized non-cooperative multiagent systems. In *Proc. International Conference on Systems, Man, and Cybernetics*, pages 1930–1935, 2012.

[26] W. Ren, R. W. Beard, and E. M. Atkins. A survey of consensus problems in multi-agent coordination. In *Proc. American Control Conference*, pages 1859–1864, 2005.

[27] C. Sammut, S. Hurst, D. Kedzier, D. Michie, et al. Learning to fly. In *Proc. 9th International Workshop on Machine Learning*, pages 385–393, 1992.

[28] R. S. Sutton and A. G. Barto. *Reinforcement learning: an introduction*, volume 1. MIT press Cambridge, 1998.

[29] U. Syed and R. E. Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in Neural Information Processing Systems*, pages 1449–1456, 2007.

[30] 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*, 75(6):1226, 1995.

[31] T. Vicsek and A. Zafeiris. Collective motion. *Physics Reports*, 517(3):71–140, 2012.

[32] C. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[33] G. M. Whitesides and B. Grzybowski. Self-assembly at all scales. *Science*, 295(5564):2418–2421, 2002.

[34] S. Zhifei and E. M. Joo. A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 5(3):293–311, 2012.

[35] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Proc. 23rd Conference on Artificial Intelligence*, pages 1433–1438, 2008.