An Adversarial Approach to Private Flocking in Mobile Robot Teams

Hehui Zheng[1], Jacopo Panerati[2], Giovanni Beltrame[2], Amanda Prorok[1]

Abstract— **Privacy is an important facet of defence against adversaries. In this letter, we introduce the problem of *private flocking*. We consider a team of mobile robots flocking in the presence of an adversary, who is able to observe all robots' trajectories, and who is interested in identifying the leader. We present a method that generates private flocking controllers that hide the identity of the leader robot. Our approach towards privacy leverages a data-driven adversarial co-optimization scheme. We design a mechanism that optimizes flocking control parameters, such that leader inference is hindered. As the flocking performance improves, we successively train an adversarial discriminator that tries to infer the identity of the leader robot. To evaluate the performance of our co-optimization scheme, we investigate different classes of reference trajectories. Although it is reasonable to assume that there is an inherent trade-off between flocking performance and privacy, our results demonstrate that we are able to achieve high flocking performance and simultaneously reduce the risk of revealing the leader.**

I. INTRODUCTION

To date, with the exception of a few recent works (e.g., [1]–[3]), the topic of *privacy* remains poorly addressed within robotics at large. Yet, privacy can be an important facet of defence against active adversaries for many types of robotics applications. Using privacy as a defence mechanism is particularly relevant for collaborative robot teams, where individual robots assume different roles with varying degrees of specialization. As a consequence, specific robots may be critical to securing the system's ability to operate without failure. Our premise is that a robot's motion may reveal sensitive information about its role within the team. To avoid threats that arise when the roles can be determined by adversaries, we need methods that ensure the anonymity of robots when their motion can be observed.

In this work, we are interested in achieving flocking behavior with *private leaders*, where privacy refers to preventing the inference of the leader's identity, based on observable motion behavior. Although classical privacy schemes, such as differential privacy, are now increasingly deployed on continuous control problems [4], they require knowledge of the output distribution of the dynamical system. Even though it is straightforward to define a basic flocking control scheme, and even sample results from its output distribution through simulation, there are no readily available analytical models that could be used to represent this distribution.

For this reason, we choose an approach towards privacy that leverages data-driven adversarial co-optimization. In

specific, we design a mechanism that optimizes flocking control parameters, such that the risk of leader inference is minimized. As the flocking performance improves, we train an adversarial discriminator that tries to infer the identity of the leader. To evaluate the performance of our co-optimization scheme, we investigate three complementary classes of reference trajectories (line, sine, and chevron).

A. Background

Flocking is a class of formation control algorithms that rely on velocity synchronization and regulation of relative distances within a group of mobile robots [5]. The aim of formation control is to drive multiple robots to satisfy certain constraints on their physical position, using local or limited information [6]. Movement in formation is crucial for a wide variety of robotics applications including surveillance [7], transportation [8], space flight control [9], and environmental monitoring [10]. The problem varies greatly depending on the desired topology of the formation, and the sensing and communication capabilities available to the robots. Given these varying constraints and the complexity of the problem, a vast number of formation control schemes have been proposed, e.g., [5], [6], [11]–[14].

Formation control schemes generally fall into three classes: *(i)* leader-follower schemes, *(ii)* virtual structure schemes, and *(iii)* behavioral formation control schemes. *Leader-follower* schemes designate one or more robots as leaders and give them access to the desired trajectory of the formation [13], [15]. The followers use local information about the leader's position and kinematics to maintain a specified offset that forms the desired formation.

In *virtual structure* schemes, the robot team is considered a single object with a designated trajectory. Each robot then uses this information in addition to local information in order to plan its own motion. These schemes often involve consensus algorithms to drive the robots' states to a common value [11]. In particular, virtual leader approaches [16]–[18] have the robots agree on the position of a virtual robot, which is then treated as a leader in some leader-follower algorithm.

Behavioral formation control schemes assign simple behaviors, such as cohesion and collision avoidance, to individual robots, with the aim of creating an emergent formation [12], [19]–[21]. A well-known example of this scheme is flocking, which is typically deployed in larger robot teams [22]. The movement of the team can be directed by a *tacit leader* [23], which is a robot that does not explicitly identify itself as a leader to other robots. Instead, it follows a reference trajectory itself, and thus, due to flock cohesion, causes the group to move with it in the desired direction.

As evident from the overview above, formation control schemes are vulnerable when they rely on robots with varying degrees of responsibility. The most obvious case is found

[1] Hehui Zheng and Amanda Prorok are with the Department of Computer Science and Technology, University of Cambridge, Cambridge, United Kingdom {hz337, asp45}@cam.ac.uk

[2] Jacopo Panerati and Giovanni Beltrame are with the Department of Software and Computer Engineering, Polytechnique Montréal, Montréal, Québec, Canada {jacopo.panerati, giovanni.beltrame}@polymtl.ca

in control schemes that leverage explicit or tacit leadership. If the leader is compromised, the entire formation can be led astray or stopped. While virtual structure approaches do not have designated leaders, it is nevertheless common for only several robots to directly receive trajectory commands, distributing this information amongst the others through local communication. This means that specific robots may act as gate-way nodes, receiving control information before other robots do. If these key robots can be identified and compromised by an adversary, the mission of the robot team can be easily disrupted.

B. Contributions

There is a dearth of research specifically tackling the problem of *role privacy* for individual robots participating in a formation control scheme. In this work, we address this gap by introducing the problem of flocking with private leaders. More specifically, we provide the following contributions:

- We formulate the problem of flocking in homogeneous mobile robot teams with private leaders.
- We develop an adversarial co-optimization method.
- We provide a study of the relation between flocking and privacy performance, and analyze the ensuing trade-off.

The following section summarizes related work. Section III introduces the problem, and Section IV presents our methodology. In Section V, we provide simulations to exhibit the behavior of the proposed co-optimization scheme, and discuss the impact on the flocking performance. Finally, Section VI concludes the article and draws directions for future research.

II. RELATED WORK

Although privacy has not traditionally featured as part of robotics research, several works have appeared in this domain in the last few years. In particular, the problem of motion planning and tracking under privacy has been a subject of focused study. In [3] and [24], the authors consider the problem of generating a target tracking policy that simultaneously preserves the target's privacy. The work deals with a powerful adversary, who is interested in computing the location of the target, and is assumed to have access to the full history of tracking information. Tsiamis et al. [25], too, consider a tracking problem. A robot is commanded to track a desired trajectory, which is transmitted through a communication channel that can be compromised by eavesdroppers. They design secure communication codes to encode the trajectory information and hide it from the eavesdroppers. In [26] and [27], attackers can spoof the sensor readings and control inputs of a robot. The authors demonstrate the existence of undetectable attacks as well as safe trajectories.

Our work differs from the aforementioned works. Compared to [25], we do not focus on the aspect of information transmission. Our adversarial model is similar to the one in [3], which assumes an adversary that has access to a history of trajectory information. Yet, we consider a distinct problem setting, where *multiple* robots are involved. In particular, our goal is to provide *role privacy*; thus, we tackle the problem of preventing an adversary from being able to distinguish the role of one robot from that of another.

The issue of role privacy was explored to some extent in prior work [28]. That work considers heterogeneous robot teams, and quantifies how easy it is for an adversary to identify the *type* of any robot in the group, based on an observation of the robot group's dynamic state. The framework, however, builds on the theory of differential privacy, and assumes the availability of an output distribution that describes the robot group's dynamic state. As previously mentioned in Section I, such a method is not easily applied to the case of flocking, where output distributions are hard to model in analytical form, and where the analytical model for the dynamical system is unknown.

Our approach towards privacy leverages an adversarial co-optimization approach [29]. The idea of adversarial privacy was recently presented by Huang et al. [30], who formulate it as a constrained minimax game between two players. Their method learns the parameters of a privatizer, which is a generative model that creates private data, and an adversarial model, which tries to infer the private variables from the output of the privatizer. Although the idea of alternating the optimization of privatizer and adversary is common in both our approaches, we apply the method to a completely different domain.

III. PROBLEM STATEMENT

Our work has two objectives: *(i)* efficient three-dimensional flocking of a team of mobile robots that follows a trajectory only known to a single leader robot; and *(ii)*, privacy of this leadership—that is, making it challenging for an artificial or human adversary to correctly identify the leader of the flock. We choose to tackle this problem using a co-optimization framework that simultaneously refines the performance of both the robot team $\mathcal{R}$ and an adversarial discriminator D. In this section, we formalize all the problem components. In Section IV, we detail their implementation.

a) Robot team and flocking models: We consider a homogeneous robot team $\mathcal{R}$ comprising of N identical robots. We refer to the position and velocity of each robot $i \in [1,..,N]$ as $\mathbf{p_i} \in \mathbb{R}^3$ and $\mathbf{v_i} \in \mathbb{R}^3$, respectively. All robots share the same limited sensing range R and each robot's neighborhood $\mathcal{N}_i^R$ is defined as the set of all robots within this radius $\mathcal{N}_i^R = \{j \in \mathcal{R} \mid r_{ij} \leq R\}$—where r_{ij} is the distance between robot i and j. We assume that each robot is capable of observing the position and velocity of all its neighbors. Thus, the observation vector o_i of robot i at time t can be written as:

$$o_i(t) = \{(\mathbf{p}(t)_j, \mathbf{v}(t)_j) \mid j \in \mathcal{N}_i^R(t)\}. \quad (1)$$

Each robot is modelled as a single integrator with velocity directly set by control $\mathbf{u}_i$. In its general formulation, the flocking control input is a function of the current state of a robot's neighborhood and a vector of parameters $\mathbf{c} \in \mathbb{R}^k$:

$$\dot{\mathbf{p}}_i(t) = \mathbf{v}_i(t) = \mathbf{u}_i(t) = f(o_i(t), \mathbf{c}). \quad (2)$$

The family of the velocity controllers for the robot team leader $l \in [1,..,N]$ is a superset of the flocking controllers in (2) as it also includes a contribution $g(\cdot)$ based on the leader's absolute position $\mathbf{p}_l$, its intended trajectory χ, and

a separate set of parameters $\mathbf{c}_l \in \mathbb{R}^w$ (with $w \geq k$):

$$\mathbf{v}_l(t) = \mathbf{u}_l(t) = f(o_l(t), \mathbf{c}) + g(\mathbf{p}_l(t), \chi, \mathbf{c}_l). \qquad (3)$$

To automate the optimization of the *flocking with leadership* behavior, we also require a formal measure of its performance loss $\mathcal{F}_{loss}$, that is a function in the form:

$$\mathcal{F}_{loss} : \mathbb{R}^{N \times 6 \times q} \longrightarrow \mathbb{R}, \qquad (4)$$

where q is the number of time steps produced by sampling $\mathbf{p_i}$ and $\mathbf{v_i}$ over T seconds at a fixed frequency $f_{\mathcal{R}}$.

b) Adversarial discriminator: The discriminator D is an adversarial agent tasked with unveiling the identity of the robot leader l of flock $\mathcal{R}$. Our assumption is that D has access to observations o_D: the positions of the entire team $\mathcal{R}$ for time windows of W seconds, that is, $o_D = \{\mathbf{p_i}(t) \mid i \in [1, .., N] \wedge t \in [t_0, t_0 + W]\}$, where t_0 is an arbitrary observation start time. Here, we assume no noise and uniform time sampling. Thus, D's role is to solve a multi-class classification problem by attaching to each observation o_D its presumed leader identifier l_D. Given discrete sampling, at fixed frequency f_D, a discriminator is any function in the form:

$$D(o_D) : \mathbb{R}^{N \times 3 \times (f_D \cdot W)} \longrightarrow [0, 1]^N. \qquad (5)$$

The co-domain $[0, 1]^N$ represents the vectors of likelihoods of each robot being the leader (see Subsection IV-C).

c) Discriminator performance and privacy: To formally define privacy as the ability to reduce the performance of an adversarial discriminator, we first define a loss function, returning non-negative penalties for each mislabelled o_D:

$$\mathcal{L}(D(o_D), y) : \mathbb{R}^N \times [1, .., N] \longrightarrow \mathbb{R}^{\geq 0}, \qquad (6)$$

where y is the correct leader identifier. The privacy loss $\mathcal{P}_{loss}$ can then be defined as a function aggregating the losses of $\hat{D}_\theta$—a discriminator implementation with parameters θ:

$$\mathcal{P}_{loss} = h(\mathcal{L}(\hat{D}_\theta(\cdot), \cdot)), \qquad (7)$$

d) Problem (adversarial private flocking): given a robot team $\mathcal{R}$ with sensing capabilities as defined in (1), *(i)* implement $\mathcal{F}_{loss}$, $\mathcal{L}$, $\mathcal{P}_{loss}$, $\hat{D}_\theta$, and *(ii)* design an adversarial co-optimization framework capable of selecting $\mathbf{c}$, $\mathbf{c}_l$ and θ such that $\mathcal{F}_{loss}$ and $\mathcal{P}_{loss}$ are simultaneously improved, through adversarial pressure, by playing the minimax game: $\arg\min_{\mathbf{c}, \mathbf{c}_l} \mathcal{F}_{loss} + \mathcal{P}_{loss}$, $\arg\max_\theta \mathcal{P}_{loss}$.

IV. METHODS

Our method comprises two components: *(i)* flocking optimization and *(ii)* leader discrimination learning. The former optimizes the controller parameters $\mathbf{c}$, $\mathbf{c}^l$ for more efficient and private flocking (i.e., minimizing $\mathcal{F}_{loss}$ and $\mathcal{P}_{loss}$), while the latter refines $\hat{D}_\theta$ to achieve higher leader-identification accuracy (i.e., increasing $\mathcal{P}_{loss}$). We co-optimize these two components in an alternating optimization procedure (similarly to [29]), as shown in Figure 1. This approach enables *(i)* the successive emergence of behaviors that adapt to adversarial pressure, and *(ii)*, the avoidance of repetitive exhaustive optimization cycles (which can be computationally prohibitive). The following text details the sub-components of our co-optimization procedure, which is finally summarized in Subsection IV-D.
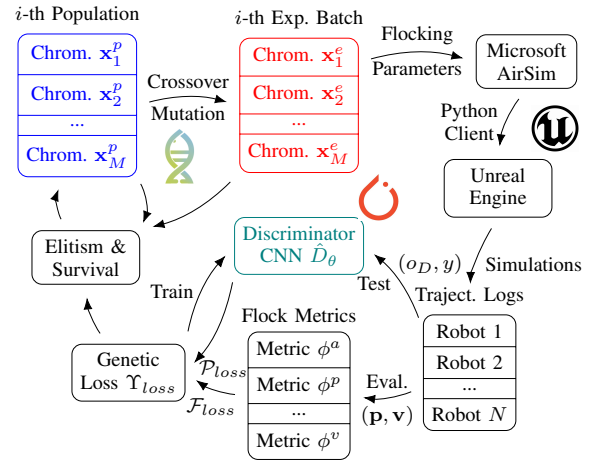

Fig. 1. Block diagram of the proposed implementation of the adversarial co-optimzation approach. From the top left, a complete, clockwise loop presents all the steps in a single generation of genetic optimization.

A. Flocking Implementation

We adopt Reynold's flocking [12], a decentralized flocking model consisting of three simple components generating control $\mathbf{u}_i(t) = \alpha^\rho \mathbf{v}_i^\rho(t) + \alpha^\sigma \mathbf{v}_i^\sigma(t) + \alpha^\tau \mathbf{v}_i^\tau(t)$. The first component $\mathbf{v}_i^\rho$ ensures collision avoidance and it is computed as:

$$\mathbf{v}_i^\rho = s\left(\frac{1}{|\mathcal{N}_i^{r^\rho}|} \sum_{j \in \mathcal{N}_i^{r^\rho}} d(r_\rho, \mathbf{p}_i, \mathbf{p}_j) \right) \qquad (8)$$

with the aid of the helper function $d(\cdot, \cdot, \cdot)$:

$$d(r^\rho, \mathbf{p}_i, \mathbf{p}_j) = \begin{cases} \frac{\mathbf{p}_i - \mathbf{p}_j}{\|\mathbf{p}_i - \mathbf{p}_j\|^2} & \text{if } 0 < \| \mathbf{p}_i - \mathbf{p}_j \| < r^\rho \\ 0 & \text{otherwise,} \end{cases}$$

where $s(\cdot)$ is a scale function to transform a position vector into a velocity. The second component $\mathbf{v}_i^\sigma$ is each robot's average neighborhood $\mathcal{N}_i^{r^\sigma}$ velocity:

$$\mathbf{v}_i^\sigma = \frac{1}{|\mathcal{N}_i^{r^\sigma}|} \sum_{j \in \mathcal{N}_i^{r^\sigma}} \mathbf{v}_j. \qquad (9)$$

To make the flocking cohesive and avoid splitting, finally, the third component $\mathbf{v}_i^\tau$ leads the robots to move towards the centre of their neighborhood $\mathcal{N}_i^{r^\tau}$:

$$\mathbf{v}_i^\tau = s\left(\frac{1}{|\mathcal{N}_i^{r^\tau}|} \sum_{j \in \mathcal{N}_i^{r^\tau}} (\mathbf{p}_j - \mathbf{p}_i) \right). \qquad (10)$$

The leader robot's trajectory tracking is implemented by a velocity component $\mathbf{v}_l^\chi$ which is returned from function $g(\cdot)$ using proportional control in the form:

$$\mathbf{v}_l^\chi(t) = \omega \frac{(\mathbf{p}_l^\chi(t) - \mathbf{p}_l)(t)}{\| \mathbf{p}_l^\chi(t) - \mathbf{p}_l(t) \|}, \qquad (11)$$

where ω is a control gain and $\mathbf{p}_l^\chi$ is the position on the trajectory at a fixed look-ahead distance. It is worth noting that the leader robot l can adopt a complete separate set of parameters, meaning that α^ρ, α^σ, α^τ, r^ρ, r^σ, and r^τ are $\in \mathbf{c}$, while α^ρ, α^σ, α^τ, r^ρ, r^σ, r^τ, and ω belong to $\mathbf{c}_l$ and can take on different values. The initial position of robot

leader l relative to the rest of the flock $\mathcal{R}$ is captured by two additional parameters i_l^x, i_l^y each $\in [-1, 1]$ and included in $\mathbf{c}_l$. Note that, while $\mathbf{c}$, $\mathbf{c}_l$ are optimized in a centralized fashion, individual robot control (2) is still based on local information only.

B. Flocking Performance Metrics and Genetic Optimization

The *flocking with leadership* model described above incorporates 15 control parameters. Optimizing such a large pool of continuous parameters cannot be done by hand or through parameter sweeping. In our proposal (Figure 1), we adopt an evolutionary approach based on classical genetic algorithms (GAs) [31], that is, optimization through a biologically-inspired stochastic search. Having defined the GA's chromosome $\mathbf{x}$ as the concatenation of $\mathbf{c}$ and $\mathbf{c}_l$, we are left with the non-trivial task of capturing its loss $\Upsilon_{loss}(\mathbf{x})$.

Let $\phi^*(t)$ represent a single performance metric at time t. We can define the sample average and variance of metrics ϕ^*, over simulation time of T samples, as follows:

$$\overline{\phi^*} = \frac{1}{T} \sum_{i=1}^{T} \phi^*(t_i) \quad \text{and} \quad \sigma^* = \frac{1}{T} \sum_{i=1}^{T} (\phi^*(t_i) - \overline{\phi^*})^2.$$

(12)

To achieve good flock alignment, the literature suggests candidate metrics—velocity correlation [22] or polarization [32]. We chose velocity correlation as it accounts for not only the direction alignment but also the magnitude similarity:

$$\phi^a(t) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} \frac{\mathbf{v}_i(t) \cdot \mathbf{v}_j(t)}{\| \mathbf{v}_i(t) \| \cdot \| \mathbf{v}_j(t) \|}.$$

(13)

This metric should be maximised to encourage alignment within the flock neighborhood (thus, we consider its negation to compute Υ_{loss}). To ensure dense but also collision-free flocking, we introduce metric ϕ^r:

$$\phi^r(t) = \frac{1}{N} \sum_{i=1}^{N} \min(r_{ij})$$

(14)

as well as metric ϕ^p:

$$\phi^p = \begin{cases} 0 & \text{if } r_- < \overline{\phi^r} < r_+ \\ \min(|\overline{\phi^r} - r_-|, |\overline{\phi^r} - r_+|) & \text{otherwise,} \end{cases}$$

(15)

where $[r_-, r_+]$ defines a range of acceptable inter-robot distances. The spacing within the flock should also be uniform. Thus, we introduce a metric for the variance of spacing among robots:

$$\phi^s(t) = \frac{1}{N} \sum_{i=1}^{N} (\min(r_{ij}) - \phi^r(t))^2.$$

(16)

To minimize the leader's tracking error to χ, we define:

$$\phi^\chi(t) = \| \mathbf{p}_l - \mathbf{p}_l^\chi \| .$$

(17)

Finally, to assess the overall flock tracking efficiency, the flock's speed is measured at the centre of mass:

$$\mathbf{v}_{\mathcal{R}} = \frac{1}{N} \left| \sum_{i=1}^{N} \mathbf{v}_i(t) \right|$$

(18)

and we defined metric ϕ^v as:

$$\phi^v = \begin{cases} 0 & \text{if } \overline{\mathbf{v}_{\mathcal{R}}} > \mathbf{v}_- \\ |\overline{\mathbf{v}_{\mathcal{R}}} - \mathbf{v}_-| & \text{otherwise,} \end{cases}$$

(19)

where $\mathbf{v}_-$ is the minimum acceptable flock speed.

We can then combine all these metrics in a vector $\mathbf{m}$:

$$\mathbf{m} = [\overline{\phi^a}, \sigma^a, \phi^p, \sigma^r, \overline{\phi^s}, \overline{\phi^\chi}, \sigma^\chi, \phi^v, \sigma^v],$$

(20)

define the hyper-parameter vector $\mathbf{b} \in \mathbb{R}^9$, and write our proposed overall flocking performance loss as $\mathcal{F}_{loss} = \mathbf{b}^T \mathbf{m}$. The GA, in turn, favours better fitness by seeking *smaller* Υ_{loss} values. Thus, in order to purely optimize for $\mathcal{F}_{loss}$ one can set $\Upsilon_{loss} = \mathcal{F}_{loss}$.

C. Adversarial Discriminator Design

Given the complexity of flocking dynamics and a lack of analytical models that can describe observations of trajectories (as defined in Section III), we opt to design the discriminator implementation $\hat{D}_\theta$ (Figure 1) using a data-driven approach. Convolutional Neural Networks (CNNs) have proven successful in multi-class classification problem for high-dimensional input with spatial information [33]. Their setup closely resembles the problem at hand of the discriminator, which is trained to distinguish the leader robot l from its followers. The discriminator's input $o_D \in \mathbb{R}^{N \times 3 \times (f_D \cdot W)}$ can be directly fed into a CNN as a multi-channel input with shape $N \times 3$ and number of channels $c = f_D \cdot W$. The CNN output is a $1 \times N$ vector stating the likelihood of each robot being the leader (see (5)). We implement (7) by first defining $\mathcal{L}$ as the multi-class cross-entropy loss of the predicted output and y, the identifier of the actual leader l:

$$\mathcal{L}(\hat{D}_\theta(o_D), y) = -\log \left(\frac{\exp(\hat{D}_\theta(o_D)[y])}{\sum_j \exp(\hat{D}_\theta(o_D)[j])} \right),$$

(21)

and, finally, privacy $\mathcal{P}$ as:

$$\mathcal{P} = \frac{1}{\sum_{\mathbf{O}} \mathcal{L}(\hat{D}_\theta(\cdot), \cdot) + \gamma}$$

(22)

where γ is a hyper-parameter and $\mathbf{O}$ a set of (o_D, y) pairs.

D. Genetic Optimization with Adversarial Training

The co-optimization of flocking $\mathcal{F}_{loss}$ and privacy $\mathcal{P}_{loss}$ is implemented through a loop that alternates genetic optimization generations and training epochs of the CNN, as shown in Figure 1. As the genetic algorithm repeatedly optimizes the controller parameters $\mathbf{c}$, $\mathbf{c}_l$, our framework also refines the parameters θ of discriminator $\hat{D}_\theta$. The aim is to generate flocking behaviors that fool the discriminator; meanwhile, the trajectories o_D and the correct label y of these improved flocking controllers are provided to the discriminator in the next optimization epoch. Online training is needed to give the discriminator stronger distinguishing ability while the GA continuously tries to beat it. We achieve this by updating the discriminator network using stochastic gradient descent for one epoch after each GA generation. To inform the GA about the current performance of $\hat{D}_\theta$, the GA's loss function
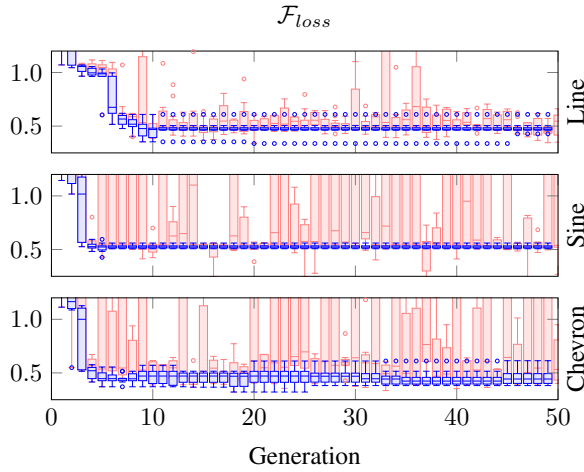
Fig. 2. Flocking performance loss $\mathcal{F}_{loss}$ through the evolution process for the 3 classes of reference trajectories. The blue box plots represent the distributions of the scores of the chromosomes in the population of the GA. The red box plots represent those of the new experiments of each generation.
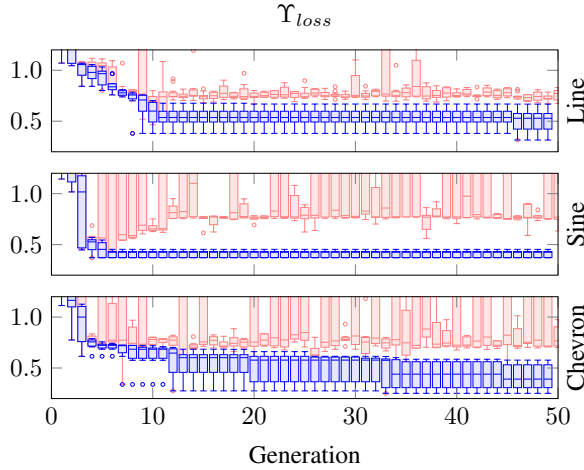

Fig. 3. Genetic loss Υ_{loss} through the evolution process for the 3 classes of reference trajectories. The blue box plots represent the distributions of the scores of the chromosomes in the population of the GA. The red box plots represent those of the new experiments of each generation.

is adapted as follows:

$$\Upsilon_{loss} = \begin{cases} \mathcal{F}_{loss} & \text{if } \mathcal{F}_{loss} \geq \kappa \\ \beta \cdot \mathcal{F}_{loss} + (1-\beta) \cdot \mathcal{P}_{loss} & \text{otherwise.} \end{cases} \quad (23)$$

Hyper-parameters κ and β can be used to discard extremely poor flocking performance and tune the trade-off between $\mathcal{F}_{loss}$ and $\mathcal{P}_{loss}$, respectively.

V. PERFORMANCE EVALUATION

In the following, we first describe our simulation setup. We then present the results and discuss our findings. All of our code is available on GitHub[1]. Additional results—including the optimized flocking parameters **c** and the generalization ability of $\hat{D}_\theta$—are also available as supplementary material[2].

[1] https://github.com/proroklab/private_flocking
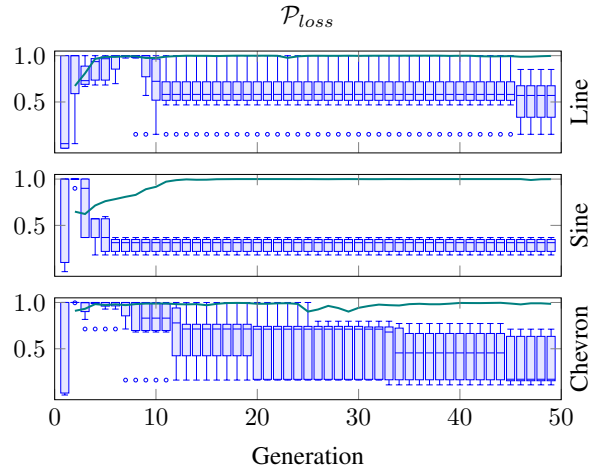[2] https://arxiv.org/abs/1909.10387

Fig. 4. Privacy performance loss $\mathcal{P}_{loss}$ through the evolution process for the 3 classes of reference trajectories. The blue box plots represent the distribution of $\mathcal{P}_{loss}$ for the solutions retained in the population of the genetic algorithm. The teal line represents the training accuracy of $\hat{D}_\theta$.

A. Simulation Setup

Our simulations were conducted using a 12-core, 3.2Ghz i7-8700 CPU and an Nvidia GTX 1080Ti GPU with 32 and 11GB of memory, respectively. Physics was provided by Unreal Engine release 4.18 (UE4). Our flocking control models (2) and (3) were implemented through the `AirSim` plugin [34] and its Python client, which conveniently exposes asynchronous APIs for the velocity control of multiple agents in custom UE4 worlds. To formalize and run the genetic evolution process, we used ESA's `pygmo` [35] library for massively parallel optimization. The discriminator's CNN implementation and training used PyTorch [36] v1.2.0, and were accelerated with Cuda v10.0 APIs.

The population size of the GA was set to $M = 10$ and evolved over 50 generations. At every generation, `pygmo` interfaced with `AirSim` and UE4 to execute 10 3-minute flock flights for each of 10 new chromosomes (experiments)—containing new values for **c**, $\mathbf{c}_l$. Specifically, we used the `SGA` algorithm with crossover and mutation probabilities of 0.9 and 0.02, respectively, a single crossover point, and elitism set to 3. Our CNN architecture included a feature extractor `Conv2d-BatchNorm2d-ReLU-MaxPool2d` followed by a three-layer `Linear-ReLU-Linear` classifier. In the feature extractor, we used kernels of size 3. Stride and zero-padding were set to 1 for both the `Conv2d` and `MaxPool2d` layers. The intermediate channel number was set to 16. The overall number of parameters in $\hat{D}_\theta$, $|\theta|$, is 0.23M. The schematic of $\hat{D}_\theta$ is available in our supplementary material[2].

We pre-trained the CNN to give the discriminator reasonable privacy performance at the beginning of the co-optimization process. We used stochastic gradient descent, for 150 epochs, and 2000 training samples from hand-tuned flocking behaviors following 3 types of leader reference trajectories $\chi \in \{line, sine, chevron\}$. The initial leader position (i_l^x, i_l^y) was uniformly distributed within the flock. After pre-training, the discriminator achieved 86.9% test accuracy on hand-tuned flocking.
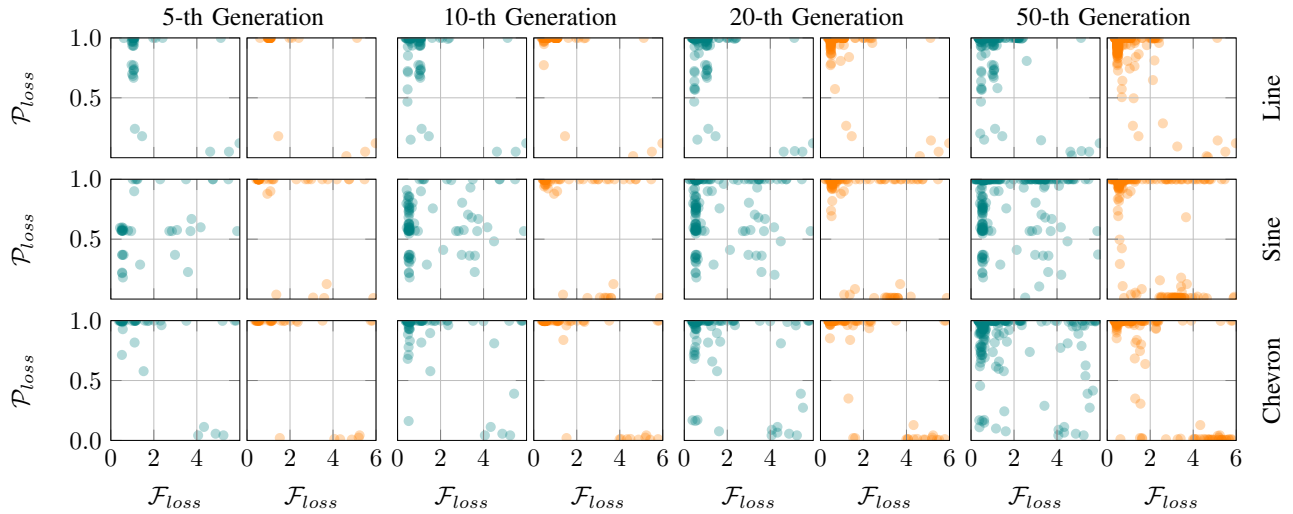
Fig. 5. Trade-off between flocking performance loss $\mathcal{F}_{loss}$ and privacy loss $\mathcal{P}_{loss}$ for the three classes of leader trajectories (the rows) at four, representative generations (5, 10, 20, and 50, i.e. the columns). The plots with the teal markers present the privacy loss $\mathcal{P}_{loss}$ returned by the discriminator CNN undergoing the online training described in Subsection IV-D. The plots with the orange markers present the privacy loss $\mathcal{P}_{loss}$ returned by the originally pre-trained network from Subsection V-A. Ideal performance is achieved in the bottom left corner of each plot.
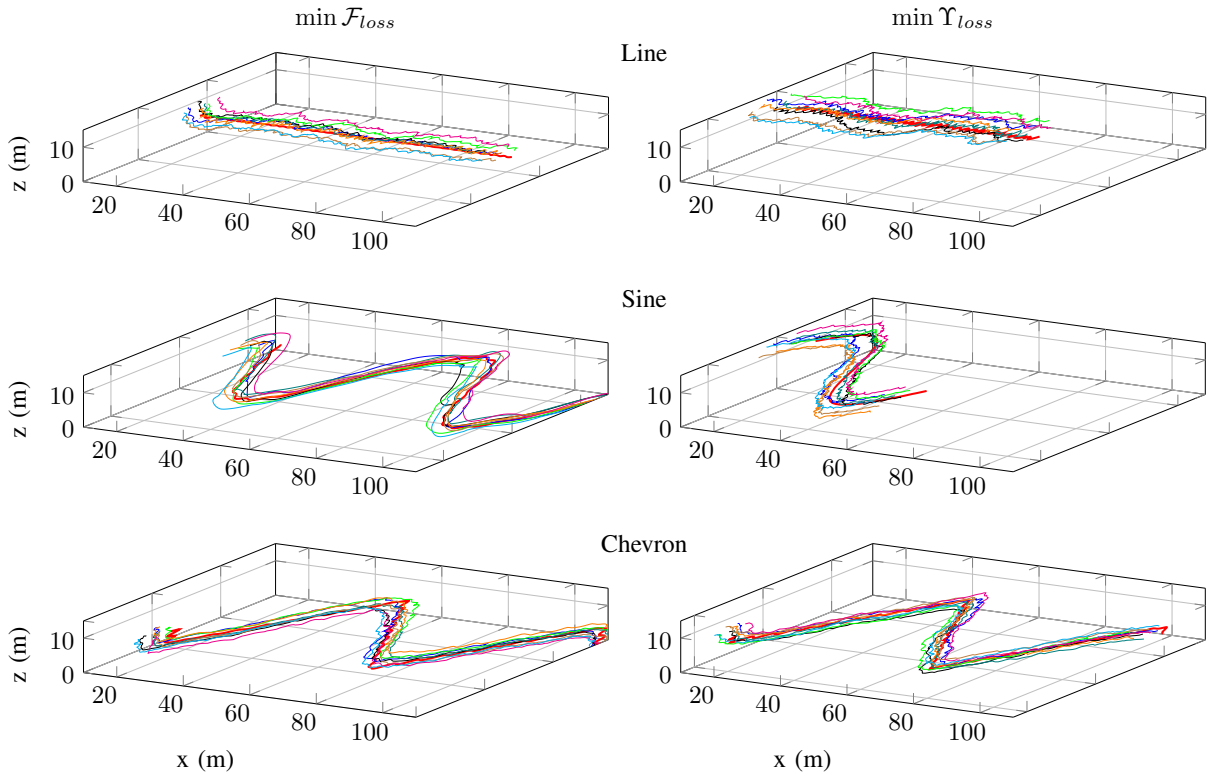
Fig. 6. The genetic champions with respect to *(i)* pure flocking performance min $\mathcal{F}_{loss}$ (left column) and *(ii)* the joint privacy/flocking fitness min Υ_{loss} (right column) for the three classes of leader trajectories, over 3-minute simulations. The trajectory trace of the leader robot is drawn in red. Note that amplitude and frequency of the sine and chevron reference trajectory are fixed at design time. Private flocks, on the sine trajectory in particular, are slower.

For the online training of the CNN—i.e., during `SGA`'s evolution—we used the 100 most recent chromosomes generated by the GA having attained a flocking loss $\mathcal{F}_{loss} \leq \kappa$. The learning rate was set to 0.025, and momentum to 0.9.

In the following subsection, we report results for 3 separate evolution processes, one for each of the trajectory types $\in \{line, sine, chevron\}$, with 9 robots (quadcopters). These

account for 1500 3-minute experiments, that is, over 3 days of simulated flight.

B. Results

Figure 2 presents the evolution—over 50 generations—of the flocking performance loss $\mathcal{F}_{loss}$ described in Subsection IV-B, for the 3 different leader reference trajectories.
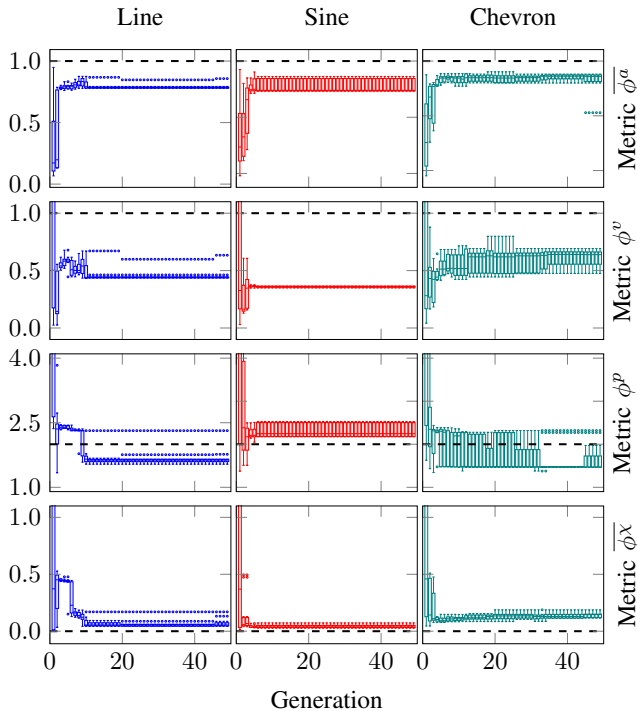
Fig. 7. Evolution of four of the individual components in the flocking loss $\mathcal{F}_{loss}$ for the three classes of leader trajectories. Each series of box plots represent the distributions of scores of the solutions in the population of the GA. The black dashed line represents the optimization target value.
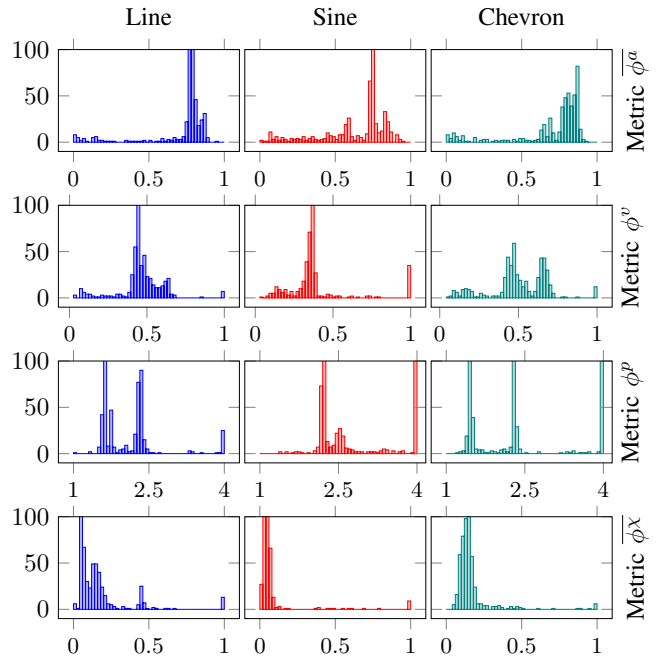

Fig. 8. Distributions over 500 experiments of the same four components of $\mathcal{F}_{loss}$ from Figure 7, for the three classes of leader trajectories.

As each generation refers to a population of M chromosomes, as well as M new, experimental chromosomes (see Figure 1), $\mathcal{F}_{loss}$ is presented by two series of 50 box plots: one (blue) describing the evolution of the distribution of $\mathcal{F}_{loss}$ for the chromosomes in the GA's population; and one (red) describing the evolution of the distribution $\mathcal{F}_{loss}$ for the experimental chromosomes. SGA quickly (within 10 generations) and effectively (for all reference trajectories) reaches good values of $\mathcal{F}_{loss}$.

The results in Figure 3 refer to the GA loss, Υ_{loss} (23). Similarly to Figure 2, Figure 3 presents information about the evolution of the chromosomes in the GA's population (in blue), and the experimental chromosomes (in red) as series of box plots. As Υ_{loss} depends on $\mathcal{F}_{loss}$, we observe similar trends between Figure 2 and 3. However, in the latter, the experimental chromosomes' performance diverges with respect to that of the GA's population, in particular for the *sine* reference trajectory. At later stages, we confront the solutions to stronger adversaries, making it more challenging for new solutions to be added to the population.

Figure 4, displays a series of box plots capturing the evolution of the distribution of the values of $\mathcal{P}_{loss}$ for the chromosomes in the population of the genetic algorithm for the 3 reference trajectories, as well as the training accuracy of the CNN $\hat{D}_\theta$. Notably, improvements in $\mathcal{P}_{loss}$ are slower than those in $\mathcal{F}_{loss}$. We also note that training accuracy of the CNN is slower to converge for the *sine* reference trajectory.

Figure 5 shows the trade-off between flocking and privacy performance. The panels in teal show how the genetic evolution successively increases the number of solutions that provide interesting trade-offs between $\mathcal{F}_{loss}$ and $\mathcal{P}_{loss}$ (bottom-left is best). The pair-wise comparison in each of

the four columns of the plot also shows that the $\mathcal{P}_{loss}$ scores provided by the pre-trained CNN (orange) are more highly polarized towards *non-private*, *good flocking* (top-left) or *private*, *poor flocking* (bottom-right).

In the left and right columns of Figure 6, we compare the trajectories generated by the *champion* chromosomes with respect to $\mathcal{F}_{loss}$ and Υ_{loss}, respectively, for each of the 3 reference trajectories. The private trajectories (on the right) tend to be slightly slower and less smooth.

Finally, Figures 7 and 8 present the evolution (over 50 genetic generations) and the overall distributions (over the 500 experiments in the 50 generations) of four of the flocking performance metrics presented in Subsection IV-B: *(i)* $\overline{\phi^a}$, quantifying the flock's alignment (1 is best); *(ii)* ϕ^v, quantifying the flock velocity (1 is best); *(iii)* ϕ^p, quantifying the flock inter-robot spacing (2 is best); and *(iv)* $\overline{\phi^\chi}$, quantifying the leader trajectory tracking error (0 is best). We see that: alignment (row 1) does well for all trajectory classes; flock velocity (row 2) is smaller for the *sine*, and varies more for *chevron*; spacing (row 3) also varies more for *chevron*; and tracking error (row 4) is smallest for *sine*.

C. Discussion

Overall, the results demonstrate that the GA is able to find very good flocking solutions, despite the large parameter space, and despite the increasing strength of the adversarial discriminator. The inclusion of privacy does not appear to harm flocking convergence, yet it is a slower process.

The final performance across the different trajectory classes is comparable, however, Υ_{loss} converges quicker for the *sine* reference than for the other two. This is corroborated by the discriminator performance, which shows a slower learning curve for the *sine*. The common denominator among *chevron* and *line* is that they are both composed of straight lines. These insights indicate increased difficulty

of hiding a leader along linear trajectories.

Although we purposefully constructed a very powerful adversary, it is not a very realistic one. Observations made from a fixed vantage point, or observations that only capture trajectories on a plane, for example, may cause the flock to move in different ways. These avenues remain to be explored in future work.

VI. CONCLUSIONS AND FUTURE WORK

This work introduced the problem of private flocking and presented a method to generate robot controllers which achieve that feat. We employed a co-optimization procedure that uses a data-driven adversarial discriminator and a genetic algorithm that optimizes flocking control parameters. Although we expected an inherent trade-off between flocking performance and privacy, our results demonstrated that we are able to achieve *both* efficient and private flocking, across different classes of reference trajectories. In this work, we considered a worst-case powerful adversary that has access to the complete, non-noisy data. Future work will consider a physical setup with real robots and a realistic adversary, who observes the robot team from specified vantage points, and through potentially noisy sensors.

REFERENCES

[1] A. Prorok and V. Kumar, "Privacy-preserving vehicle assignment for mobility-on-demand systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1869–1876.

[2] L. Li, A. Bayuelo, L. Bobadilla, T. Alam, and D. A. Shell, "Coordinated multi-robot planning while preserving individual privacy," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2188–2194.

[3] Y. Zhang and D. A. Shell, "Complete characterization of a class of privacy-preserving tracking problems," *The International Journal of Robotics Research (IJRR)*, vol. 38, no. 2-3, pp. 299–315, 2019.

[4] S. Han and G. J. Pappas, "Privacy in control and dynamical systems," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 309–332, 2018.

[5] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, March 2006.

[6] K.-K. Oh, M.-C. Park, and H.-S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.

[7] D. Van der Walle, B. Fidan, A. Sutton, C. Yu, and B. D. Anderson, "Non-hierarchical uav formation control for surveillance tasks," in *American Control Conference (ACC)*. IEEE, 2008, pp. 777–782.

[8] J. Bom, B. Thuilot, F. Marmoiton, and P. Martinet, "A global control strategy for urban vehicles platooning relying on nonlinear decoupling laws," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2005, pp. 2875–2880.

[9] R. W. Beard, J. Lawton, and F. Y. Hadaegh, "A coordination architecture for spacecraft formation control," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 6, pp. 777–790, 2001.

[10] S. Li, Y. Guo, and B. Bingham, "Multi-robot cooperative control for monitoring and tracking dynamic plumes," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 67–73.

[11] W. Ren and N. Sorensen, "Distributed coordination architecture for multi-robot formation control," *Robotics and Autonomous Systems*, vol. 56, no. 4, pp. 324 – 333, 2008.

[12] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '87. New York, NY, USA: ACM, 1987, pp. 25–34.

[13] J. Desai, J. Ostrowski, and V. Kumar, "Modeling and control of formations of nonholonomic mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 905–908, 2001.

[14] B. T. Fine and D. A. Shell, "Unifying microscopic flocking motion models for virtual, robotic, and biological flock members," *Autonomous Robots*, vol. 35, no. 2, pp. 195–219, Oct 2013.

[15] D. Gu and Z. Wang, "Leader?follower flocking: Algorithms and experiments," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1211–1219, Sep. 2009.

[16] N. E. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials and coordinated control of groups," in *IEEE Conference on Decision and Control*, vol. 3. IEEE, 2001, pp. 2968–2973.

[17] M. Egerstedt, X. Hu, and A. Stotsky, "Control of mobile platforms using a virtual vehicle approach," *IEEE Transactions on Automatic Control*, vol. 46, no. 11, pp. 1777–1782, Nov 2001.

[18] H. Su, X. Wang, and Z. Lin, "Flocking of multi-agents with a virtual leader," *IEEE Transactions on Automatic Control*, vol. 54, no. 2, pp. 293–307, Feb 2009.

[19] F. Cucker and S. Smale, "Emergent behavior in flocks," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 852–862, May 2007.

[20] A. E. Turgut, H. Çelikkanat, F. Gökçe, and E. Şahin, "Self-organized flocking in mobile robot swarms," *Swarm Intelligence*, vol. 2, no. 2, pp. 97–120, Dec 2008.

[21] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Transactions on Tobotics and Automation*, vol. 14, no. 6, pp. 926–939, 1998.

[22] G. Vásárhelyi, C. Virágh, G. Somorjai, T. Nepusz, A. E. Eiben, and T. Vicsek, "Optimized flocking of autonomous drones in confined environments," *Science Robotics*, vol. 3, no. 20, 2018.

[23] S. A. Amraii, P. Walker, M. Lewis, N. Chakraborty, and K. Sycara, "Explicit vs. tacit leadership in influencing the behavior of swarms," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2209–2214.

[24] J. M. O?kane, "On the value of ignorance: Balancing tracking and privacy using a two-bit sensor," in *Algorithmic Foundations of Robotics VIII*. Springer, 2009, pp. 235–249.

[25] A. Tsiamis, A. B. Alexandru, and G. J. Pappas, "Motion planning with secrecy," in *2019 American Control Conference (ACC)*, July 2019.

[26] G. Bianchin, Y. Liu, and F. Pasqualetti, "Secure navigation of robots in adversarial environments," *IEEE Control Systems Letters*, vol. 4, no. 1, pp. 1–6, Jan 2020.

[27] Y.-C. Liu, G. Bianchin, and F. Pasqualetti, "Secure trajectory planning against undetectable spoofing attacks," *Automatica*, vol. 112, p. 108655, 2020.

[28] A. Prorok and V. Kumar, "A macroscopic privacy model for heterogeneous robot swarms," in *Swarm Intelligence*, M. Dorigo, M. Birattari, X. Li, M. López-Ibáñez, K. Ohkura, C. Pinciroli, and T. Stützle, Eds. Cham: Springer International Publishing, 2016, pp. 15–27.

[29] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680.

[30] C. Huang, P. Kairouz, X. Chen, L. Sankar, and R. Rajagopal, "Context-aware generative adversarial privacy," *CoRR*, vol. abs/1710.09549, 2017.

[31] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine Learning*, vol. 3, no. 2, pp. 95–99, Oct 1988.

[32] C. Gershenson, A. Muoz-Melndez, and J. L. Zapotecatl, "Performance metrics of collective coordinated motion in flocks," in *Artificial Life Conference Proceedings 13*. MIT Press, 2016, pp. 322–329.

[33] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, May 2010, pp. 253–256.

[34] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017.

[35] "esa/pagmo2: pagmo 2.11.1," Aug. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3364433

[36] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

This section contains the supplementary material mentioned in Section V and presented through Figures 9, 10, 11, 12, and 13.
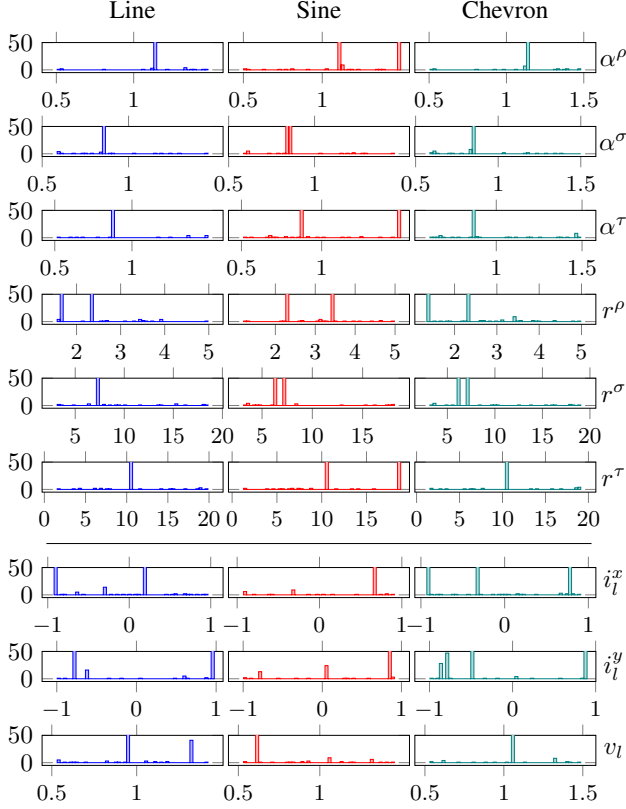

Fig. 9. Density distributions of the flocking parameters in $\mathbf{c}$, $\mathbf{c}_l$ (see Subsection IV-A). The distributions' peaks indicate the most common choices made by the GA throughout the co-optimization process. The optimal flocking parameters in $\mathbf{c}$—top six rows—are relatively consistent across trajectories; the optimal relative initial position and velocity of the leader—bottom three rows—vary with the choice of reference trajectory. The preferred leader starting position has a lateral offset (i_l^y close to 1 or -1), especially for line and chevron trajectories. On the other hand, the co-optimized leader's velocity v_l for the sine trajectory is the slowest.
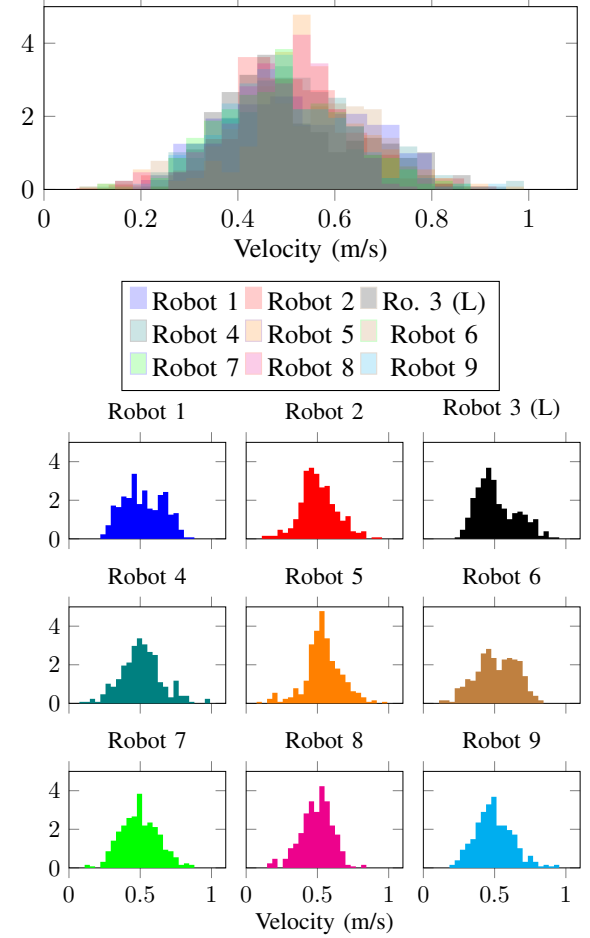

Fig. 10. Density distributions of the velocities of all robots (followers and leader) in the flock, for a 3' co-optimized line trajectory. We note from the histogram with superimposed distributions (top chart) that the individual distributions are indistinguishable. Thus, a first-principles-based discriminator would be unable to identify the leader—this difficulty being exacerbated by the fact that only ∼3% of the data shown in the histogram actually goes into each input vector o_D of $\hat{D}_\theta$ (i.e., to the observer). Even if one could generate trajectories that elicit clear differences between these distributions, the controller's ability to evolve could then learn how to invalidate a fixed, model-based discriminator.
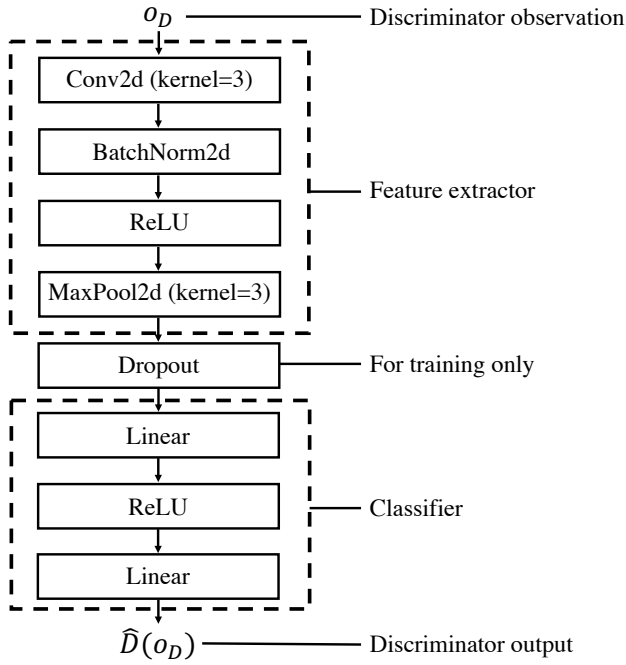
Fig. 11. This figure presents the schematic of $\hat{D}_\theta$'s CNN architecture—as described in Subsection V-A. The third dimension of input o_D (see (5)) is a finite number of channels $f_D \cdot W$. In our implementation, we considered f_D of $\sim$2Hz and experimented with W between 1 and 15 seconds, before fixing it to 5"—a value that offered an appropriate trade-off between training accuracy and generalization. While o_D only contains the robots' positions over time window W, we reckon that—in case the robots' velocities were relevant to the discriminator's task—$\hat{D}_\theta$ would infer them by simple derivation.
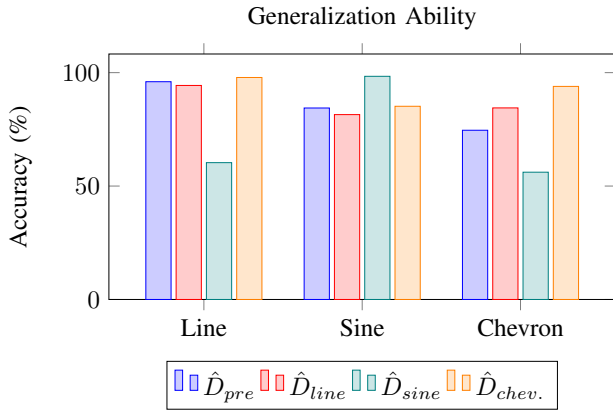

Fig. 12. We performed additional experiments to evaluate the ability of a CNN trained on specific trajectories to generalize towards others. $\hat{D}_{pre}$ is pre-trained on a combination of the three trajectories while $\hat{D}_{line}$, $\hat{D}_{sine}$, and $\hat{D}_{chev}$ are trained during co-optimization—each on a specific trajectory. We test these four discriminators against optimized, but newly generated, trajectories of all three types. We observe that the optimized discriminators can achieve on-par or better performance when tested against the specific trajectory they were trained on. While $\hat{D}_{line}$ generalizes well (better than $\hat{D}_{pre}$) to the chevron trajectory and vice versa, $\hat{D}_{sine}$ achieves worse performance than $\hat{D}_{pre}$ against the other two trajectories. The good generalization between line and chevron trajectories is likely explained by the fact that the typical distance travelled by the flock within the discriminator observation window W is about 3 m. Thus, for chevron trajectories composed of $\sim$30-meter segments, the observed data is still mostly straight line-like. Nonetheless, we note that, since the classification problem is not binary, $\hat{D}_{sine}$ still performs much better than random guessing, even on line and chevron trajectories.
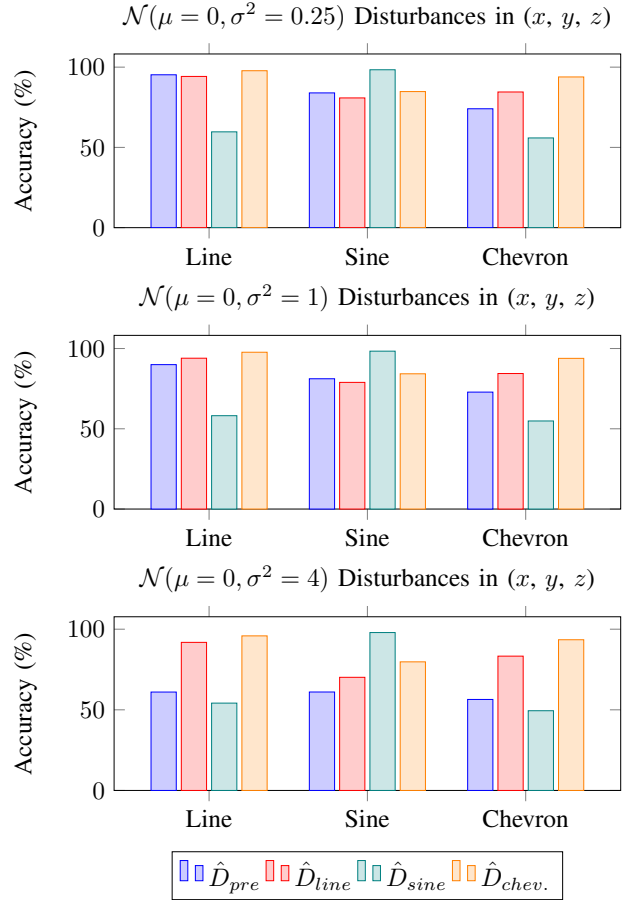

Fig. 13. We expanded the set of experiments from Figure 12 to feed our CNNs with test inputs including artificial disturbances. Normally distributed, zero-mean, fixed-variance disturbances were added in the (x, y, z) observations of the discriminators. The disturbance variances $[0.25, 1, 4]$ were chosen to resemble the typical travel distance within one observation time window W (3 m) and the average flocking spacing (1 to 5 m). For $\sigma^2 = 0.25$ and $\sigma^2 = 1$, performance only slightly degrades, suggesting that the discriminators are, in fact, moderately robust to noise. Even though the discriminators' accuracy decrease for $\sigma^2 = 4$, when compared to $\hat{D}_{pre}$, we observe that the co-optimized discriminators fare much better. In fact, the effect of bounded disturbances on the difficulty of leader identification appears to be limited and partially mitigated by the additional training.