Memory-Based Resilient Control Against Non-cooperation in Multi-agent Flocking

Mingyue Zhang
School of Computer and Information Science, Southwest University
Chongqing, China
myzhangswu@swu.edu.cn

Nianyu Li✉
ZGC Lab
Beijing, China
li_nianyu@pku.edu.cn

Jialong Li
Waseda University
Tokyo, Japan
lijialong@fuji.waseda.jp

Jiachun Liao
Nanhu Lab
Zhejiang, China
jliao@nanhulab.ac.cn

Jiamou Liu
University of Auckland
Auckland, New Zealand
jiamou.liu@auckland.ac.nz

ABSTRACT

Inspired by natural flocking behaviors, researchers aim to develop a distributed control approach for artificial agents to mimic these behaviors. The main challenge lies in maintaining the resilience of the artificial flock, as some agents inevitably display non-cooperative behavior, thereby deviating from the flocking objective. Existing control approaches, especially those based on learning algorithm, are susceptible to forgetting issues that non-cooperative agents can exploit to disrupt the flock formation. To address this problem, this study introduces a memory-based resilient control approach that strategically analyzes historical data across three distinct time scales (long, short, and periodic). The implementation of a long short periodic-term memory (LSP) algorithm employs accumulative discounted credibility evaluated by Q-learning to recognize long-term non-cooperation, utilizes a filtering rule to establish a trusted set excluding short-term non-cooperation, and integrates fast Fourier transform to refine the trusted set against periodic inconsistency. We assess the effectiveness of this approach through extensive experiments. The results highlight the potential and advantages of using LSP in flocking, enhancing the resilience of multi-agent flocking against complex non-cooperative threats.

KEYWORDS

Flocking; Multi-agent System; Collective Behaviour; Resilient Consensus Control

ACM Reference Format:
Mingyue Zhang, Nianyu Li✉, Jialong Li, Jiachun Liao, and Jiamou Liu. 2024. Memory-Based Resilient Control Against Non-cooperation in Multi-agent Flocking. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 10 pages.

Nianyu Li (li_nianyu@pku.edu.cn) is corresponding author. This work is supported by the Fundamental Research Funds for the Central Universities (No. SWU-KQ23005), and the youth project of science and technology research program of Chongqing Education Commission of China (No. KJQN20230).

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

The natural flocking behaviors refer to the collective behavior of a large number of individuals (e.g., birds, fish, and insects), forming a group where the distance between every pair of individuals remains within an appropriate range (i.e., separation & cohesion) and maintaining motion consensus (i.e., alignment) through local information exchange [37]. Over the years, the natural phenomenon has attracted widespread attention in the academic community, leading to various theories and methods for explaining or simulating this collective behavior [27, 44], which also have greatly inspired research on the multi-agent motion control. Utilizing flocking control, a collection of artificial agents (e.g., drones, mobile robots) can accomplish tasks that cannot be performed by individual agents alone, such as patrol, joint military operations and search-and-rescue applications [14, 16, 42, 45].

One of the pivotal challenges in designing the flocking control approach lies in addressing the so-called threat of *non-cooperative behaviors*. In practical scenarios, the running environments of agents are inevitably unpredictable and potentially hostile (e.g., energy exhausting, malicious attack); consequently, certain agents may not conform to the prescribed control law (e.g., random motion, motion with constant speed) [35]. Behaviors that contravene the prescribed control law, thereby violating the separation, cohesion and alignment objective of flocking, are termed non-cooperative behaviors, and agents exhibiting such behaviors are known as non-cooperative agents [21, 39]. As reported by much of the literature [21, 40], existing control approaches without resilience mechanisms are highly vulnerable, as even a single non-cooperative agent can provoke catastrophic effects on the entire control process (e.g., the failure of the flocking, deviation from the intended course).

To enhance the resilience of multi-agent flocking under non-cooperation, solutions have been proposed and categorized into three main classes: *outlier-based*, which discards certain outliers within the data received [24, 40]; *trustworthiness-based*, where agents evaluate the trustworthiness of their neighbors through the analysis of history and disregard data from untrustworthy agents [30, 33]; *learning-based*, which leverages learning algorithm to adapt agents' behaviors [20, 21]. However, the first two solutions have limitations, such as strict assumptions on non-cooperative agents' number and specific communication topologies among agents, and additional computational overhead, hindering practical application [46]. The third solution, known for its wide adaptability, is susceptible to catastrophic forgetting issues [5]. This vulnerability can be exploited by non-cooperative agents, particularly those we term as *inconsistent* agents. An inconsistent behavior is marked by strategic shifts between cooperation and non-cooperation. For example, such an agent might gain trust through temporary cooperation, then abruptly switch to non-cooperation to create instability, only to later cooperate again to restore confidence. Hence, the flocking behavior will be severely disrupted by the inconsistent agents.

In this paper, we investigate the resilience of learning-based multi-agent flocking control. Recognizing the challenge posed by catastrophic forgetting in conventional learning-based solutions, we propose a novel *memory-based resilient control* approach to tolerate the different non-cooperative behaviors (especially from inconsistent agents), ensuring the remaining cooperative agents achieve flocking formation. We have designed the *long short periodic-term memory* (LSP) control algorithm for each cooperative agent. The LSP algorithm is capable of automatically detecting and isolating non-cooperative agents, adjusting the motion of cooperative ones to ensure the flocking formation is maintained. This involves: (1) Long-Term Analysis: Learning the accumulative discounted credibility with Q-learning to identify neighbors violating separation, cohesion or alignment over long time scales. (2) Short-Term Focus: Designing a filtering rule to create a trusted set that excludes neighbors exhibiting non-cooperative behavior over short time scales. (3) Periodic Inspection: Incorporating fast Fourier transform (FFT) into the examination of neighbors' history, refining the trusted set by excluding neighbors demonstrating periodic inconsistency.

This three-tiered approach provides a sophisticated understanding of non-cooperation and allows for precise interventions. Moreover, to thoroughly assess the applicability and effectiveness of our proposed solutions across a diverse range of non-cooperation scenarios, we rigorously define the behavior model for non-cooperative agents. Our experimental evaluations demonstrate that this memory-extended solution successfully safeguards multi-agent flocking control against the multi-dimensional threat of non-cooperation across various time scales, outperforming existing approaches.

The main contributions are summarised as follows. (1)*A Sophisticated Non-Cooperation Model*: This non-cooperative model, by introducing two strategies of switching between cooperation and non-cooperation, significantly enhances the camouflage and destructiveness of non-cooperation, which allows the non-cooperative agents to bypass existing resilience mechanisms and disrupt flocking behavior. (2) *Memory-Based Resilient Control Approach*: By analyzing the agent's historical data across three time scales, our approach identifies the traits of non-cooperative agents. This overcomes the inherent forgetting issues in learning-based flocking formation, strengthening the resilience against such non-cooperative behaviors. (3) *Algorithm Implementation*: We integrate Q-learning, filtering rules, and FFT into the LSP, crafting an algorithm that offers resilient control against non-cooperation.

2 RELATED WORK

Given a set of agents, they are in a multi-agent flock formation if the distance between every pair of agents is "neither too large nor too small" (i.e., separation & cohesion) and the motion of all agents maintain as much consistency as possible (i.e., alignment) [10, 37]. Various methods for controlling the "artificial flock" such as drones or mobile robots have been proposed, including well-known algorithms like the boids model [41], manipulation the flock through adding influencing agents [17], leader-follower flocking control [34], the leader-follower control enhanced by deep reinforcement learning [49]. These methods primarily assume that all agents are entirely controlled by a homogeneous control algorithm.

With the growing application of multi-agent flocking control in an open world characterized by uncertainties and potential maliciousness, the resilience of the control is attracting increasing attention. An agent's non-cooperative behavior is among the factors that most significantly affect the resilience of control. Such behavior may lead to the the failure of the flocking [13, 38, 44]. [44] proposes a method based on graph topological properties for resilient control. [23] designs a data spoofing strategy to interfere with their intended flight paths, but does not provide a defense mechanism. [7] leverages Graph Neural Networks to counteract fundamental attacks on the communicated information in flocking. More generally, the theoretical foundation for the resilient flocking control is multi-agent consensus control(MACC). MACC conceptualizes agents' physical metrics (e.g., position, velocity, and acceleration) as "characteristics", aiming to study a distributed control method to achieve consensus among the agents in these characteristics [32]. Based on the current research on MACC, the primary solutions for resilience (some of which are explicitly designed for flocking, while others require modifications to be applied to flocking) can be roughly classified into three categories. It is crucial to emphasize that the resilient flocking or MACC primarily examines how the normal agents should react to the failure or malicious attack on other agents, ensuring these normal ones maintain the control objectives. They do not consider how to prevent the agents from failure or being attacked.

Outlier-based methodologies, discarding the outliers within the data received from their neighbors, represent the first category of resilience solutions. They are implemented either temporarily or permanently, i.e., discarding outliers for the current time or permanently discarding the data generated by the agents that have previously generated outliers. Some typical algorithms include: MSR [24] assumes a priori knowledge of the maximum number (F_{nc}) of non-cooperative agents, and temporarily eliminates the F_{nc} largest values and F_{nc} smallest values. [25] introduced the weights of each agent into MSR for eliminating outliers more accurately. [40] identifies non-cooperative agents and also ensures that the convergence value remains within the initial values of cooperative agents. Essentially, these algorithms exhibit proficiency in convergence and robustness; nonetheless, they are limited by network connectivity or the requirement to know F_{nc}, which hinders their application in general cases [46, 50].

The second category involves a series of *trustworthiness-based* methodologies, wherein the trustworthiness of each agent is evaluated based on history, subsequently leading to the permanent exclusion of the most untrustworthy agents. Some typical algorithms include: RoboTrust [33] computes the trustworthiness by employing observations and conducting statistical inferences from diverse historical perspectives, retaining solely the most credible agents for synchronization purposes. [30] extended RoboTrust into the second-hand evidence. [6] further enhanced the robustness of RoboTrust by designing various local decision rules based on local evidence. [4] concentrated on non-cooperation within colonies and designed a pheromone-based coordination (where the pheromone is a form of trustworthiness). Nonetheless, these methods raise storage and computational requirements and do not consider cooperation among non-cooperative agents [21].

The third category encompasses a suite of *learning-based* methodologies, which adapt agents' behaviors through using learning algorithms. Some typical algorithms include: [21] considered the flocking as a multi-agent consensus control problem, and introduced MADDPG [31] to learn a control policy for cooperative agents. Additionally, they designed a Q-learning based algorithm for updating agents' actions. [20] designed a multi-armed bandit sampling to isolate non-cooperative agents. [38] introduced mean field game for controlling the large-scale agents, and used reinforcement learning to solve the game. [48] studied the nonlinear strict-feedback-dynamic multi-agent consensus control, and proposed a leader-following control based on deep reinforcement learning. Nonetheless, the core of these algorithms is reinforcement learning, which inevitably suffers from catastrophic forgetting issues [5]. Non-cooperative agents are likely to exploit this characteristic, continuously adjusting their behaviors to deceive the algorithm and ultimately disrupt the flocking objective. In the field of federated learning, there have been prior efforts to design attacks based on the core idea of inconsistent agents, such as adaptive attack [12] and 3DFed [26]. However, in the research on flocking, no one has yet considered this type of non-cooperative behaviors.

In addition, some studies focus on the aspects of communication, multi-agent system architecture, and multi-agent motion planning. The typical works include: [1] employed formal methods to ensure the safety, reachability, and other attributes of control methods during the flocking process. [2] proposed a verification approach for the communication of multiple agents in partially-known environments. [3] modeled the flocking as a graphical game to analyze the equilibrium states that agents can achieve in self-interest scenarios. [29] focused on the application in the microgrids systems, transformed the consensus control into a zero-sum differential game, but it cannot be directly applied to flocking yet.

3 PROBLEM FORMULATION

We begin by defining the dynamic model of agents. This sets the stage for the analysis and formulation of the system. Next, we detail the objective of multi-agent flocking control, outlining the goals that the system is expected to achieve. Finally, we introduce a linear aggregation policy, which forms the cornerstone of our approach.

3.1 Agent Dynamic Model

In this paper, the motion of the agents is a second-order dynamic model, wherein each agent adjusts its velocity by altering its acceleration, thereby changing its state.

Definition 1. *The dynamic model for all agents is formulated as a 8-tuple $\mathcal{M} = \langle N, E, t, S, V, A, \omega, \mathcal{T} \rangle$, where: (1) N is a finite set of n agents labeled by $1, 2, ..., n$; (2) $E \subseteq N \times N$ is a finite set of directed edges where an edge (i, j) signifies agent i can send information to agent j; E is called the communication topology for agents; (3) $t \in \{0, 1, ..., T\}$ represents discrete time-steps with T as the maximum time step; (4) $S = S_1 \times S_2 \times ... \times S_n$ is a joint state space, with $S_i \subseteq \mathbb{R}^m$ being the state space of agent i, and m is the dimension of the state; $\vec{s}(\in S) = [s_1, ..., s_n]$ is the joint state of all agents, where $s_i \in S_i$ is agent i's state; (5) $V = V_1 \times V_2 \times ... \times V_n$ is a joint velocity space, with $V_i \subseteq \mathbb{R}^m$ as the velocity space of agent i and $\vec{v}(\in V) = [v_1, ..., v_n]$ as the joint velocity of all agents, where $v_i \in V_i$ is agent i's velocity; (6) $A = A_1 \times A_2 \times ... \times A_n$ is a joint acceleration space, with $A_i \subseteq \mathbb{R}^m$ as the acceleration space of agent i and $\vec{a}(\in A) = [a_1, ..., a_n]$ as the joint acceleration of all agents, where $a_i \in A_i$ is agent i's acceleration; (7) $\omega \in \Omega$ represents bounded noise induced by environment, where $\Omega \subseteq \mathbb{R}^m$ is the noise value range; (8) $\mathcal{T} : S_i \times V_i \times \Omega \to S_i$ is a state dynamic function, and $\mathcal{T}(s, v, \omega) = s'$ represents that if agent i takes velocity v in state s under environmental noise ω, the next state will transition into s'.* □

As in previous works [13, 15], we utilize the following discrete-time dynamics function for each agent i in the system:

$$s_{i,t+1} = \mathcal{T}(s_{i,t}, v_{i,t}, \omega_{i,t}) = s_{i,t} + v_{i,t} + \omega_{i,t}, \forall i \in N \tag{1}$$

where $s_{i,t}$, $v_{i,t}$, $\omega_{i,t}$ denote the state, action, and environmental noise for agent i at time step t, respectively. The system's initial state $\vec{s}_0 = [s_{1,0}, s_{2,0}, ..., s_{n,0}]$ is arbitrarily determined.

The velocity dynamics is:

$$v_{i,t+1} = v_{i,t} + a_{i,t}, \forall i \in N \tag{2}$$

where $a_{i,t}$ denotes the acceleration for agent i at time step t.

In terms of communication, $\mathcal{N}_i$ comprises agents that can send information to agent i defined as $\mathcal{N}_i = \{j | (j, i) \in E\}$ while $\mathcal{N}_{out,i}$ comprises agents that can receive information from agent i defined as $\mathcal{N}_{out,i} = \{i | (i, j) \in E\}$.

3.2 Control Objective and Control Policy

The agents are categorized into two groups: cooperative and non-cooperative. The characteristics of these cooperative agents are described below. The two objectives of the flocking, i.e., separation & cohesion, and alignment, are formulated as follows.

Definition 2 (separation-cohesion). *The distance between any two cooperative agents should fall within a reasonable range, i.e.,*

$$\forall i, j \in N^c . \forall t \in \{0, 1, ..., T\} . \delta_{low} \le \left\| s_{j,t} - s_{i,t} \right\|_2 \le \delta_{up} \tag{3}$$

where δ_{low} and δ_{up} are the lower and upper bounds of the state distance, respectively, $0 \le \delta_{low} < \delta_{up}$, and N^c represents the set of all cooperative agents. □

Definition 3 (alignment). *After a certain time step T_{con}, the velocities of all agents will converge to the same value, i.e.,*

$$\forall t > T_{con} . \max_{i,j \in N^c} \left\| v_{j,t} - v_{i,t} \right\|_2 < \delta_v \tag{4}$$

where $\delta_v \ge 0$ is a threshold of the post-convergence velocity. □

In this paper, the control objective is to have all cooperative agents adjust their acceleration to modify their states, satisfying Eq.(3) and Eq.(4) conditions. We ground the agents' policies on a *boids* control law [41]. Specifically, for agent $i \in N^c$, the policy is defined as $\pi_i : S_i \times S_{\mathcal{N}_i} \times V_i \times V_{\mathcal{N}_i} \to A_i$, where S_i/V_i is agent i's state/velocity space, $S_{\mathcal{N}_i}/V_{\mathcal{N}_i}$ is the joint state/velocity space of agents in $\mathcal{N}_i$. The policy is given by:

$$\pi^{\mathrm{L1}}(s_{i,t}, s_{j,t}) = \gamma_1 \Big(\frac{\delta_{\mathrm{up}} + \delta_{\mathrm{low}}}{2} - \left\| s_{j,t} - s_{i,t} \right\|_2 \Big) \frac{(s_{i,t} - s_{j,t})}{\left\| s_{j,t} - s_{i,t} \right\|_2} \tag{5}$$

$$\pi^{\mathrm{L2}}(v_{i,t}, v_{j,t}) = \gamma_2(v_{j,t} - v_{i,t}) \tag{6}$$

$$\pi_i(s_{i,t}, \vec{s}_{\mathcal{N}_i,t}, v_{i,t}, \vec{v}_{\mathcal{N}_i,t}) = \sum_{j\in\mathcal{N}_i} \alpha_{i,j,t}\big(\pi^{\mathrm{L1}}(s_{i,t}, s_{j,t}) + \pi^{\mathrm{L2}}(v_{i,t}, v_{j,t})\big) \tag{7}$$

where π^{L1} is designed for achieving the flocking objectives defined in Definition 2, and when $\frac{\delta_{\mathrm{up}}+\delta_{\mathrm{low}}}{2} > \|s_{j,t} - s_{i,t}\|_2$, the policy produces a velocity away from agent j for i, and vice versa produces a velocity close to j (cf. [43]). π^{L2} is designed for achieving the flocking objectives defined in Definition 3, and it consistently reduces the velocity distance between any two agents (cf. [21]). γ_1 and γ_2 are scaling factors. $\alpha_{i,j,t}$ is set to $\frac{1}{|\mathcal{N}_i|}$. $s_{i,t}/v_{i,t}$ and $\vec{s}_{\mathcal{N}_i,t}/\vec{v}_{\mathcal{N}_i,t}$ denote the state/velocity of agent i at t and the joint state/velocity of agent i' neighbors at t, respectively. π_i is homogeneous for the cooperative agents, and it is represented for π.

At last, in alignment with other studies [21, 22], we make the following assumption to ensure information exchange among cooperative agents throughout this paper.

Assumption 1. *Given a dynamic model $\mathcal{M}$, represented by its topology $G = \langle N, E\rangle$, we assume that $G[N^c]$ is fixed and rooted. Meanwhile, the complementary part $G - G[N^c]$ can be arbitrary. Here, $G[N']$ is defined as the graph $\langle N', E'\rangle$ where $E' = \{(i,j) | i \in N', j \in N', (i,j) \in E\}$, and $G - G[N']$ is the graph $\langle N'', E''\rangle$ with $N'' = \{i | i \in N, i \notin N'\}$ and $E'' = \{(i,j) | (i,j) \in E, (i,j) \notin E'\}$.* □

4 NON-COOPERATIVE BEHAVIORS

This section defines a non-cooperative behavior model and details the implementation of specific instances of non-cooperation.

4.1 Behavior Model for Non-cooperative Agents

The non-cooperative agents in a flock deviate from the prescribed control policy, potentially hindering the achievement of separation-cohesion and alignment. Their behavior model is defined as follows.

Definition 4. *$\mathcal{B} = \langle N^{nc}, \mathcal{F}, \rho\rangle$, where: (1) $N^{nc} = \{i \in N | i \notin N^c\}$ represents the set of non-cooperative agents; (2) $\mathcal{F} = \{\mathcal{F}_i | i \in N^{nc}\}$ is a set of tampering functions, where $\mathcal{F}_i : S_i \times S_{\mathcal{N}_i} \times V_i \times V_{\mathcal{N}_i} \to A_i$ is the tampering function of agent i; (3) $\rho : \{0, 1, ..., T\} \times N^{nc} \to \{0, 1\}$ is a role function that determines whether the non-cooperative agents violate the prescribed control policy π. For each non-cooperative agent i and time step $t \in \{0, 1, \cdots, T\}$, $\rho(t, i) = 0$ indicates that i follows π at time step t, while $\rho(t, i) = 1$ indicates a violation.* □

Definition 5. *Given a behavior model $\mathcal{B} = \langle N^{nc}, \mathcal{F}, \rho\rangle$ and a cooperative agent's policy π, the policy for non-cooperative agent $i \in N^{nc}$ is given as follows:*

$$\psi_i(s_{i,t}, \vec{s}_{\mathcal{N}_i,t}, v_{i,t}, \vec{v}_{\mathcal{N}_i,t}) = \begin{cases} \pi(s_{i,t}, \vec{s}_{\mathcal{N}_i,t}, v_{i,t}, \vec{v}_{\mathcal{N}_i,t}), \text{ if } \rho(t,i) = 0 \\ \mathcal{F}(s_{i,t}, \vec{s}_{\mathcal{N}_i,t}, v_{i,t}, \vec{v}_{\mathcal{N}_i,t}), \text{ if } \rho(t,i) = 1 \end{cases}$$ □

4.2 Implementation of Non-cooperation

We consider the following four types of tampering functions, which are very destructive in multi-agent system [21, 22, 24, 33]. (1) **Constant acceleration** (CA), i.e., $\mathcal{F}_i(\cdot,\cdot,\cdot,\cdot) = \#\mathrm{const}$, where #const is a constant that does not change over time. (2) **Random acceleration** (RA), i.e., $\mathcal{F}_i(\cdot,\cdot,\cdot,\cdot) = \mathrm{random}(t)$, where $\mathrm{random}(t) \in A_i$ returns a random variable at each time step. (3) **Random velocity** (RV), i.e., $\mathcal{F}_i(\cdot,\cdot,\cdot,\cdot) = \mathrm{random}(t) - v_{i,t}$. According to Eq.(2), the velocity dynamics function becomes the following form: $v_{i,t+1} = v_{i,t} + \mathcal{F}_i(s_{i,t}, \vec{s}_{\mathcal{N}_i,t}) = v_{i,t} + \mathrm{random}(t) - v_{i,t} = \mathrm{random}(t)$. (4) **Sign reversal** (SR), i.e. $\mathcal{F}_i(\cdot,\cdot,\cdot,\cdot) = -\pi(\cdot,\cdot,\cdot,\cdot)$.

As for role function, we consider the following two types.

(1) **Byzantine** role function, i.e.,

$$\rho(t,i) = \begin{cases} 0, & \text{if } uni(t) \geq \beta_i \\ 1, & \text{otherwise} \end{cases} \tag{8}$$

where $uni(t)$ returns a variable satisfying a uniform distribution between 0 and 1, β_i is a probability, and β_i is called Byzantine ratio.

(2) **Inconsistent** role function: the design of it is more complex and may have various implementation ways, such as modeling agents using non-cooperative game theory [3] or designing learning-based non-cooperative behavior [21]. This paper draws inspiration from the attack method of 3DFed [26], one of the most potent attack methods in federated learning, and presents the following inconsistent role function.

$$\rho(t,i) = \begin{cases} 0, & \text{if } \frac{\sum_{k=1}^{t}\sum_{j\in\mathcal{N}_i}\mathbb{I}(ind(k,i))}{t|\mathcal{N}_i|} \geq \phi \\ 1, & \text{otherwise} \end{cases} \tag{9}$$

where $ind(k,i) = (\|v_{i,k-1} - v_{j,k-1}\|_2 < \|v_{i,k} - v_{j,k}\|_2) \vee (\delta_{\mathrm{low}} > \|s_{j,t} - s_{i,t}\|_2) \vee (\|s_{j,t} - s_{i,t}\|_2 > \delta_{\mathrm{up}})$ is a condition, $\mathbb{I}(ind(k,i))$ is the indicator function that is 1 when $ind(k,i)$ is true or 0 otherwise. The core idea of this function is that non-cooperative agents continuously monitor the state/velocity dynamics of their neighbors to assess whether they are trusted by their neighbors. Once the untrustworthiness value is higher than ϕ, the agent will exhibit π behavior; otherwise, it will exhibit $\mathcal{F}$ behavior.

Byzantine role function have been mentioned in the relevant literature [9, 19, 51] and is a common threat in the multi-agent system. To the best of our knowledge, inconsistent role function is addressed for the first time in this paper, which characterises the oscillation behavior between cooperation and non-cooperation.

5 MEMORY-BASED RESILIENT CONTROL

This section outlines a memory-based resilient control approach, and details the control algorithm to counter the non-cooperation.

5.1 Approach Overview

Fig. 1 illustrates the approach overview, emphasizing a distributed method where agents rely solely on neighbor information, eliminating the need for a centralized controller. The control loop for an agent $i \in N^c$ consists of five activities: (1) *Memory Storing*: Records and analyzes neighbor state history. (2) *LSP Controller*: Utilizes neighbor history to determine Q values of agent i's neighbors and a trusted set, addressing non-cooperation across varying time scales. (3) *Weight Update*: Modifies the weights of i concerning its neighbors using Q values and the trusted set. (4) *Acceleration and velocity Update*: Adjusts i's $a_{i,t}$ and $v_{i,t}$ using Eq.(2) and Eq.(7). (5) *State Update*: Update i's state using Eq.(1). Each cooperative agent applies these activities at every time step to control its state, subsequently transmitting the state and velocity to agents in $\mathcal{N}_{out,i}$.

5.2 Q-learning Based Long-Term Memory

For each cooperative agent i, the long-term memory mechanism can learn credibility (i.e., $Q_{i,j,t}$) for each neighbor j. For $j \in \mathcal{N}_i$

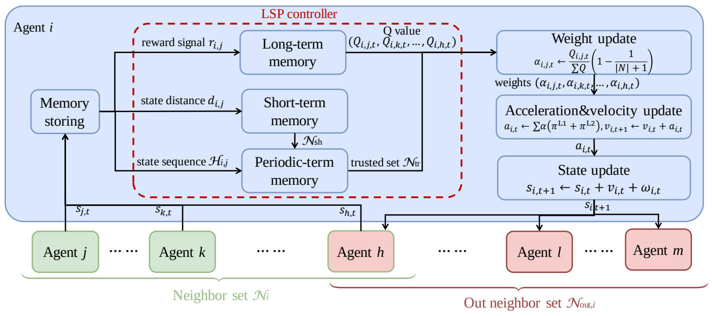
Agent i
LSP controller
reward signal $r_{i,j}$
Long-term memory
Q value $(Q_{i,j,t}, Q_{i,k,t}, \dots, Q_{i,h,t})$
Weight update $\alpha_{i,j,t} \leftarrow \frac{Q_{i,j,t}}{\sum Q}\left(1-\frac{1}{|N|+1}\right)$
weights $(\alpha_{i,j,t}, \alpha_{i,k,t}, \dots, \alpha_{i,h,t})$
Memory storing
state distance $d_{i,j}$
Short-term memory
Acceleration&velocity update $a_{i,t} \leftarrow \sum\alpha(\pi^{L1}+\pi^{L2}), v_{i,t+1} \leftarrow v_{i,t}+a_{i,t}$
$\mathcal{N}_{sh}$
$a_{i,t}$
state sequence $\mathcal{H}_{i,j}$
Periodic-term memory
trusted set $\mathcal{N}_{tr}$
State update $s_{i,t+1} \leftarrow s_{i,t}+v_{i,t}+\omega_{i,t}$
$s_{j,t}$
$s_{k,t}$
$s_{h,t}$
$s_{i,t+1}$
Agent j
Agent k
Agent h
Agent l
Agent m
Neighbor set $\mathcal{N}_i$
Out neighbor set $\mathcal{N}_{out,i}$

Figure 1: Overview of the proposed approach

with a higher credibility, the adjacent weights (i.e., $\alpha_{i,j,t}$) are larger, and for any neighbor with a lower credibility, $\alpha_{i,j,t} \longrightarrow 0$ as $t \longrightarrow \infty$.

First, we define a reward function for the flocking as follows.

$$\begin{aligned} r_{i,j,t} &= f\left(\left|\frac{\delta_{\text{up}}+\delta_{\text{low}}}{2} - \left\|s_{j,t}-s_{i,t}\right\|_2\right| + \left\|v_{j,t}-v_{i,t}\right\|_2\right) \\ &= e^{-\lambda_t\left(\left|\frac{\delta_{\text{up}}+\delta_{\text{low}}}{2}-\|s_{j,t}-s_{i,t}\|_2\right|+\|v_{j,t}-v_{i,t}\|_2\right)} \end{aligned} \tag{10}$$

where $\lambda_t > 0$ is a scaling factor that increases with time t. The first term ($\left|\frac{\delta_{\text{up}}+\delta_{\text{low}}}{2} - \left\|s_{j,t}-s_{i,t}\right\|_2\right|$) addresses separation&cohesion, penalizing deviations from the optimal distance between agent pairs (Definition 2). The second term ($\left\|v_{j,t}-v_{i,t}\right\|_2$) focuses on alignment, rewarding minimal velocity differences between agents (Definition 3). Eq. (10) ensures the reward ranges between 0 and 1. A reward close to 1 indicates optimal agent alignment and spacing, while it nears 0 for significant discrepancies.

Second, the long-term memory mechanism employs an iterative approach akin to Q-learning, where the credibility is designed as the cumulative discounted sum of rewards.

$$Q_{i,j,t} = Q_{i,j,t-1} + \eta_t(r_{i,j,t} - Q_{i,j,t-1}) \tag{11}$$

where η_t is a learning rate decreasing with time step. It indicates that as the time step increases, the impact of past rewards on the current credibility progressively diminishes.

5.3 Trusted Set Generation

When a cooperative agent updates its acceleration through Eq.(7), instead of using information from $\mathcal{N}_i$, it only uses the information from the agents in the trusted set (denoted as $\mathcal{N}_{tr}(i) \subset \mathcal{N}_i$). $\mathcal{N}_{tr}(i)$ is generated as follows.

5.3.1 *Short-term memory for generating the trusted set.*

Let $i \in N^c$, and $j \in \mathcal{N}_i$. If j is cooperative, the velocity distance between i and j is expected to satisfy the following conditions (cf. [18, 20]):

$$d_{i,j}(t) - d_{i,j}(t-1) \le 0 \tag{12}$$

where $d_{i,j}(t) = \left\|v_{j,t} - v_{i,t}\right\|_2$, and $\|\cdot\|_2$ denotes the 2-norm.

We use Eq.(12) as a basic indicator for revealing whether a neighbor behaves cooperatively or not. Hence, we design a candidate set $\mathcal{N}_{sh}(i) = \{j | j \in \mathcal{N}_i, d_{i,j}(t) - d_{i,j}(t-1) \le 0\}$.

The short-term memory is a *tit-for-tat* (TFT) [36] in nature. An agent with short-term memory will first cooperate, then subsequently replicate the neighbors' previous decision (cooperation/non-cooperation). A non-cooperative agent with inconsistent role function can make this defense ineffective. It can exhibit cooperative behavior to gain the trust of its neighbors, and then abruptly switch to non-cooperation to destabilize its neighbors.

5.3.2 *Periodic-term memory for refining the trusted set.*

In response to the threat of inconsistent agents, we propose a periodic-term memory for the refinement of $\mathcal{N}_{sh}(i)$.

Let $i \in N^c$, and $j \in \mathcal{N}_i$. $\mathcal{H}_{i,j}^T = [d_{i,j}(0), \dots, d_{i,j}(T)]$ is a velocity distance sequence up to time step T. We use an exponential function, denoted as $\hat{d}_{i,j}(t)$, to fit $\mathcal{H}_{i,j}^T$. The residual distance sequence is $\Delta\mathcal{H}_{i,j}^T = [\varepsilon_{i,j}(0), \dots, \varepsilon_{i,j}(T)]$, where $\varepsilon_{i,j}(t) = \hat{d}_{i,j}(t) - d_{i,j}(t)$. Since the inconsistent role function is periodic, $\varepsilon_{i,j}(t)$ should also be periodic. It is well known that Fourier transform can analyze the frequency and amplitude of periodic signals. Hence, we use Fast Fourier Transform[11] to generate the frequency spectrum of $\Delta\mathcal{H}_{i,j}^T$. The spectrum of $\Delta\mathcal{H}_{i,j}^T$ is denoted as $\text{FFT}(\varepsilon_{i,j}) : Fr \longrightarrow Am$, where Fr/Am is the frequency/amplitude space.

Then, the indicator for revealing whether a neighbor is inconsistent is designed as follows.

$$\max_{x \in Fr} \text{FFT}(\varepsilon_{i,j})(x) \le ||\omega||_2 \tag{13}$$

where $||\omega||_2$ represents the maximum value of the noise. If Eq.(13) dose not hold, then i will remove j from $\mathcal{N}_{sh}(i)$. The refined trusted set of agent i is defined as follows:

$$\mathcal{N}_{tr}(i) = \{j | j \in \mathcal{N}_{sh}(i), \max_{x \in Fr} \text{FFT}(\varepsilon_{i,j})(x) \le ||\omega||_2\} \tag{14}$$

Subsequently, we delve into the rationale behind the selection of exponential functions for fitting $\mathcal{H}_{i,j}^T$. If agent j is cooperative, $d_{i,j}(t)$ can be expanded as follows:

$$\begin{aligned} d_{i,j}(t) = \|&(v_{j,t-1} - v_{i,t-1}) + \sum_{k \in \mathcal{N}_j} \alpha_{j,k,t-1}(v_{k,t-1} - v_{j,t-1}) \\ &- \sum_{k \in \mathcal{N}_i} \alpha_{i,k,t-1}(v_{k,t-1} - v_{i,t-1})\|_2 \end{aligned} \tag{15}$$

For simplicity of analysis, we assume that $\mathcal{N}_i \cup \{i\} = \mathcal{N}_j \cup \{j\}$, and the weights for all agents are the average weights, i.e., $\forall j, k, t, \alpha_{j,k,t} = \alpha = \frac{1}{|\mathcal{N}_i|}$. Thus, the second and third terms of Eq.(15) can be rewritten in the following form:

$$\begin{aligned} &\sum_{k \in \mathcal{N}_j} \alpha_{j,k,t-1}(v_{k,t-1} - v_{j,t-1}) - \sum_{k \in \mathcal{N}_i} \alpha_{i,k,t-1}(v_{k,t-1} - v_{i,t-1}) \\ &= \alpha \sum_{k \in \mathcal{N}_j/\{i\}} (v_{k,t-1} - v_{j,t-1}) + \alpha(v_{i,t-1} - v_{j,t-1}) \\ &- \alpha \sum_{k \in \mathcal{N}_i/\{j\}} (v_{k,t-1} - v_{i,t-1}) - \alpha(v_{j,t-1} - v_{i,t-1}) \\ &= \alpha(|\mathcal{N}_i| + 1)(v_{i,t-1} - v_{j,t-1}) = (1+\alpha)(v_{i,t-1} - v_{j,t-1}) \end{aligned} \tag{16}$$

where $\mathcal{N}_i/\{j\} = \{k \in \mathcal{N}_i | k \neq j\}$. Drawing upon the outcomes derived from Eq.(16), we can rewrite Eq.(15) as an exponential function in terms of t, as presented below.

$$\begin{aligned} d_{i,j}(t) &= \left\|(v_{j,t-1} - v_{i,t-1}) + (1+\alpha)(v_{i,t-1} - v_{j,t-1})\right\|_2 \\ &= \left\|\alpha(v_{i,t-1} - v_{j,t-1})\right\|_2 = \alpha d_{i,j}(t-1) = \alpha^t d_{i,j}(0) \end{aligned}$$

If j is inconsistent, based on Eq.(9), $d_{i,j}(t)$ is:

$$d_{i,j}(t) = \begin{cases} \alpha d_{i,j}(t-1), & \text{if } \frac{\sum_{k=1}^{t}\sum_{j \in \mathcal{N}_i} \mathbb{I}(ind(k,i))}{t|\mathcal{N}_i|} \ge \phi \\ \mathcal{F}_i(\cdot,\cdot,\cdot,\cdot), & \text{otherwise} \end{cases} \tag{17}$$

Despite the potential complexity of the time-variant signal $d_{i,j}(t)$, Eq.(17) provides ample justification for the inference that exponential components and periodic fluctuations are inherent within this signal. Consequently, if we eliminate the exponential portion of the signal (i.e., $\alpha d_{i,j}(t-1)$), the residual signal should manifest significant periodicity. This underpins our decision to employ exponential functions for data fitting, and the use of FFT for signal spectrum analysis.

Algorithm 1: LSP for cooperative agent i

```
1  Initialize $\alpha_{i,j,0} = \frac{1}{|\mathcal{N}_i|}$, $Q_{i,j,0} = 1, \forall j \in \mathcal{N}_i$;
2  $\mathcal{H}_{i,j} \leftarrow \emptyset, \forall j \in \mathcal{N}_i$;
3  for $t \leftarrow 1$ to $T$ do
4      Receive states and velocities of $i$'s neighbors $s_{j,t}, v_{j,t}, \forall j \in \mathcal{N}_i$;
5      $\mathcal{N}_{sh}(i) \leftarrow \emptyset$, $\mathcal{N}_{tr}(i) \leftarrow \emptyset$;
6      for $j \in \mathcal{N}_i$ do
7          $d_{i,j,t} \leftarrow \|v_{j,t} - v_{i,t}\|_2$, and put $d_{i,j,t}$ into $\mathcal{H}_{i,j}$;
           // long-term memory
8          Compute $r_{i,j,t}$ with Eq.(10);
9          $Q_{i,j,t} \leftarrow Q_{i,j,t-1} + \eta_t(r_{i,j,t} - Q_{i,j,t-1})$;
           // short-term memory
10         if $d_{i,j,t} < d_{i,j,t-1}$ then
11             $\mathcal{N}_{sh}(i) \leftarrow \{j\} \cup \mathcal{N}_{sh}(i)$;
12         end
13     end
       // periodic-term memory
14     for $j \in \mathcal{N}_{sh}(i)$ do
15         $\hat{d}(t) \leftarrow$ curve_fit$(\mathcal{H}_{i,j})$;
16         $y \leftarrow$ FFT$(\mathcal{H}_{i,j} - \hat{d}(t))$;
17         if $\max_{x \in Fr} y(x) \le \|\omega\|_2$ then
18             $\mathcal{N}_{tr}(i) \leftarrow \{j\} \cup \mathcal{N}_{tr}(i)$;
19         end
20     end
21     for $j \in \mathcal{N}_{tr}(i)$ do
22         $\alpha_{i,j,t} \leftarrow \frac{Q_{i,j,t}}{\sum_{j \in \mathcal{N}_{tr}(i)} Q_{i,j,t}}(1 - \frac{1}{|\mathcal{N}_{tr}(i)|})$;
23     end
24     $a_{i,t} \leftarrow \sum_{j \in \mathcal{N}_{tr}(i)} \alpha_{i,j,t}(\pi^{L1}(s_{i,t}, s_{j,t}) + \pi^{L2}(v_{i,t}, v_{j,t}))$;
25     Update $v_{i,t+1}$ using Eq.(2), and update $s_{i,t+1}$ using Eq.(1);
26 end
```

5.4 LSP Algorithm

The LSP algorithm, detailed in Alg.1, is divided into three key components: (1)*Long-term Memory* (Lines 8-9) implements memory across extended timeframes, utilizing the reward function defined in Eq.(10). (2)*Short-term Memory* (Lines 10-12) generates the candidate set $\mathcal{N}_{sh}(i)$, focusing on recent state/velocity dynamics. (3)*Periodic-term Memory* (Lines 14-19) employs an exponential fitting function $\hat{d}(t) = a * e^{-b*t} + c$. Line 15 calculates the fitting parameters a,b, and c through "curve_fit", generating the fitted function $\hat{d}(t)$. Lines 16 and 17 apply FFT to generate the frequency spectrum $\mathcal{H}_{i,j} - \hat{d}(t)$, identifying the peak value with $\max_{x \in Fr} y(x)$.

It should be noted that the function fitting operation in line 15 may be time-consuming with an increasing number of data points. A sampling approach across time steps can be employed to select fewer data points for fitting, thereby reducing the computational time. Additionally, when time steps are small and data is insufficient, the periodic-term memory (Lines 14-20) will remain inactive initially, omitting execution until adequate data is accumulated.

Table 1: Different policies for non-cooperative agents

Tampering function	Role function		
	-	Inconsistent	Byzantine
Constant acceleration (CA)	CA	ICA	BCA
Random acceleration (RA)	RA	IRA	BRA
Random velocity (RV)	RV	IRV	BRV
Sign reversal (SR)	SR	ISR	BSR

6 EVALUATION

This section delineates the evaluations designed to answer the following research questions: (1) What is the effectiveness of various resilient control approaches in defending against non-cooperation? (2) How do the number of non-cooperative agents and environmental noise affect various resilient control approaches?

6.1 Experimental Setup

Non-Cooperative Policies Implementation. Based on the previously described behavior model, we establish 12 non-cooperative agent policies, detailed in Table 1. For CA, #const is randomly set at the beginning of each experiment (i.e., $t = 0$) and remains fixed. For RA/RV, random(t) returns a uniform distribution between minimum and maximum acceleration/velocity. In addition, the second column of Table 1(CA/RA/RV/SR) defines cases where functions are set as $\phi = 0$ (for inconsistency) or $\beta_i = 1$ (for Byzantine). The dimension of state/velocity/acceleration is set to two.

Communication Networks. In the experiments, two types of graphs are adopted as the communication topology for the multi-agent system: random graph (generated at each experiment's beginning) and large-scale network (email-Eu-core temporal network[1]). All graphs comply with Assumption 1.

Comparison Approaches and Hardware. Four compared approaches are considered in our experiments, including QC [21], a learning-based approach; Linear [8, 51],i.e., $\alpha_{i,j,t} = \frac{1}{|\mathcal{N}_i|}$, a fundamental multi-agent data aggregation method; CBFDI [28], employing Gossip for fault detection and isolation; and MADDPG [31], a baseline multi-agent reinforcement learning for continuous control, similar to the design presented in [21]. The experiments are conducted on an Ubuntu 16.04.01 desktop equipped with Intel(R) Core(TM) i7-7700 CPU@3.6GHz and an Nvidia Quadro P600 GPU.

6.2 Results and Analysis

State dynamics. We first examined the changes in agent states over time during the flocking process. Fig. 2 displays the results of our approach versus four other comparison methods under four typical non-cooperative behaviors. The communication network is set to random graph. For inconsistent role functions, $\phi = 0.2$; for Byzantine, $\beta_i = 0.2$. The x-axis of Fig. 2 represents time steps, while the y-axis depicts the average state difference between two cooperative neighboring agents. The two dashed lines at the bottom of the figure represent δ_{up} and δ_{low}, and they are set to 0.05 and 0.25, respectively. The results indicate that, in the presence of non-cooperative behaviors, our approach ensures the state difference between neighboring agents consistently remains within $[\delta_{\text{up}}, \delta_{\text{low}}]$. In contrast, other methods deviate significantly from this $[\delta_{\text{up}}, \delta_{\text{low}}]$ under certain non-cooperative behaviors. For instance, under IRV,

[1]Data from SNAP: http://snap.stanford.edu/data/

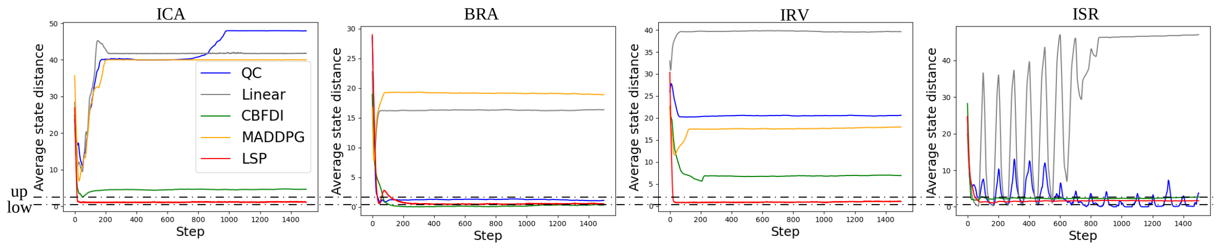

Figure 2: The dynamics of states on random network ($|N^c| = 7$, $|N^{nc}| = 3$.)

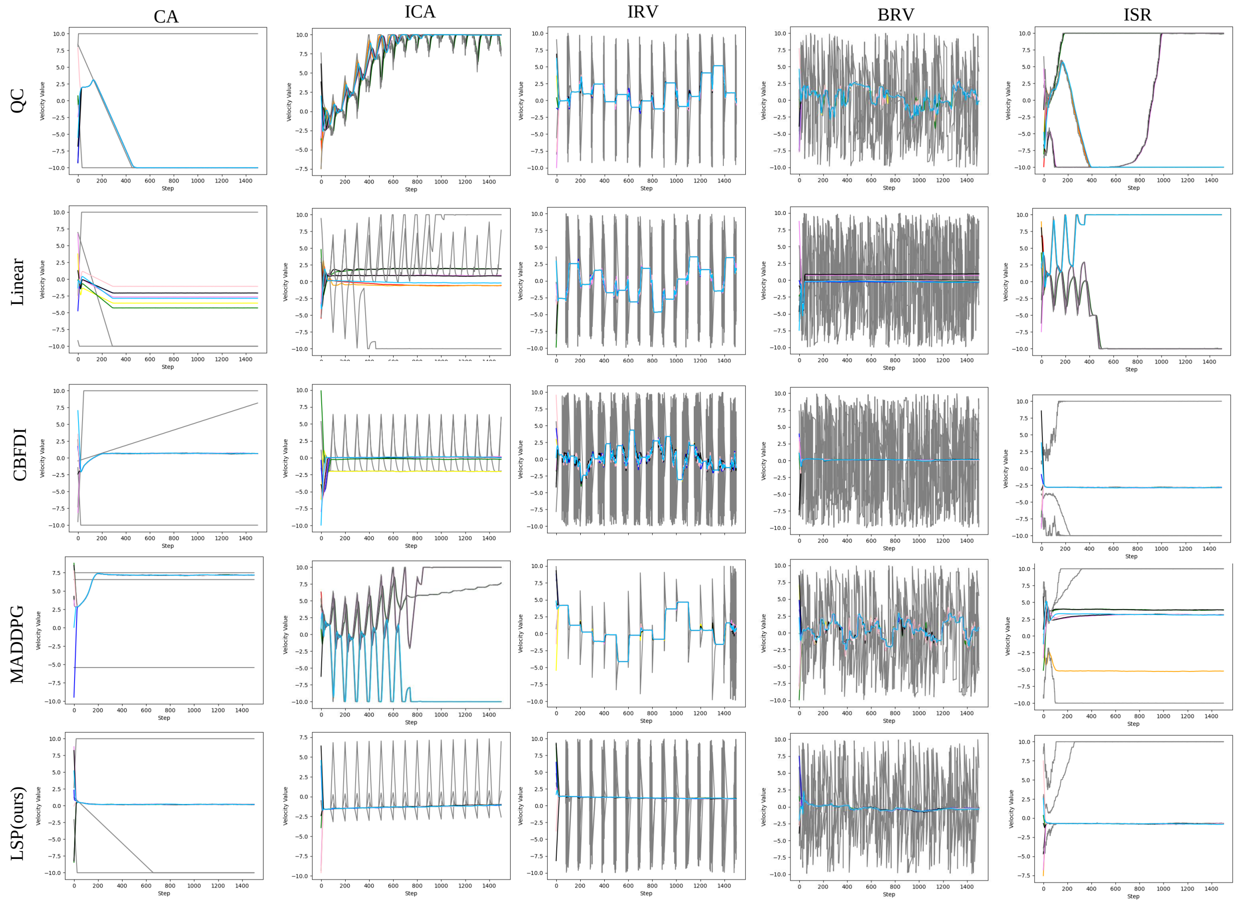

Figure 3: The dynamics of velocities on random network ($|N^c| = 7$, $|N^{nc}| = 3$.)

the state differences for QC, Linear, MADDPG, and CBFDI exceed δ_{up} by a wide margin. Under BRA, CBFDI's state difference falls below δ_{low}, whereas MADDPG and Linear exceed δ_{up} significantly.

Velocity dynamics. Fig. 3 represents the time step (x-axis) versus one dimension of the velocities (y-axis), with colored curves denoting cooperative agents. The gray curves are the non-cooperative agents' velocity dynamics. Fig. 3 reveals: (1) LSP: It enables cooperative agents to converge swiftly and smoothly to a fixed velocity, and once converged, they remain stable and are no longer influenced by non-cooperative agents. (2) Other approaches: They display varying degrees of influence from non-cooperation, including the drastic variation in the velocity of cooperative agents with the dynamics of non-cooperative agents' velocities (e.g., QC/Linear/CBFDI/MADDPG under IRV) or the convergence of cooperative agents to entirely distinct velocities (e.g., Linear/CBFDI under ICA).

Success rate & convergence time steps. A successful flocking is defined as all cooperative agents satisfying the constraints of Definition 2 and 3 after 500 time steps. The convergence time step is the number of steps required for velocity convergence in a successful run. The communication topology is set to a large-scale

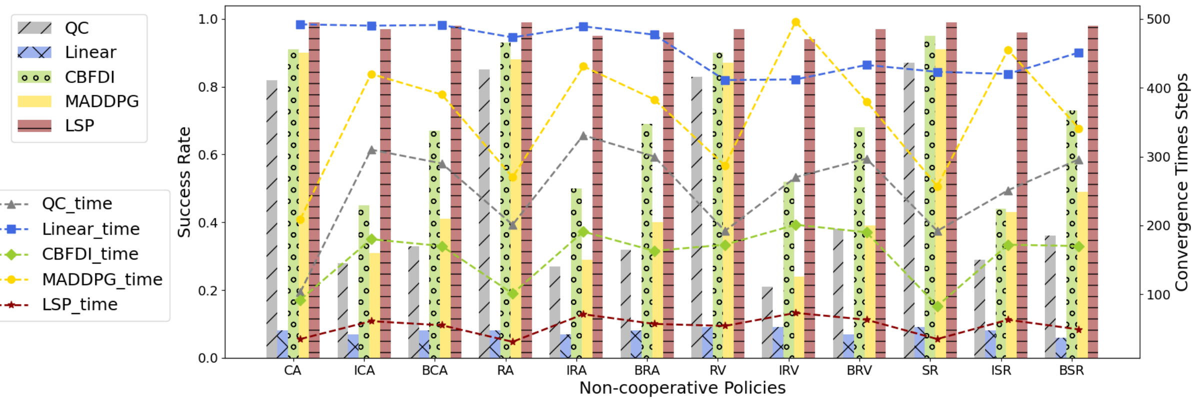

Figure 4: Comparison of success rates and convergence time steps of 5 approaches with 12 types of non-cooperation.

Table 2: Comparison of various approaches under different numbers of non-cooperative (IRV) agents.

$\frac{\lvert N^{nc}\rvert}{\lvert N\rvert}$	QC	Linear	CBFDI	MADDPG	LSP(ours)
0.00	0.98, 45	0.98, 51	0.96, 49	0.96, 55	0.98, 50
0.01	0.84, 75	0.70, 82	0.90, 82	0.96, 69	0.98, 54
0.05	0.80, 102	0.56, 132	0.88, 99	0.90, 87	0.98, 57
0.10	0.56, 143	0.24, 168	0.68, 103	0.60, 193	0.96, 64
0.20	0.42, 193	0.10, 298	0.60, 158	0.46, 391	0.96, 66
0.30	0.22, 270	0.08, 412	0.52, 201	0.24, 496	0.94, 73
0.40	0.10, 336	0.08, 484	0.28, 289	0.22, 498	0.86, 95
0.50	0.10, 354	0.06, 499	0.26, 318	0.18, 498	0.84, 102

Table 3: Comparison of various approaches under different degrees of environmental noise ($\delta_v = 0.3$ cf. Definition 3).

$\lvert\lvert\omega\rvert\rvert_2$	QC	Linear	CBFDI	MADDPG	LSP(ours)
$0.0\ \delta_v$	0.38, 246	0.10, 404	0.60, 198	0.42, 412	0.96, 68
$0.2\ \delta_v$	0.22, 270	0.10, 412	0.52, 201	0.24, 496	0.94, 73
$0.4\ \delta_v$	0.20, 310	0.08, 427	0.48, 265	0.20, 499	0.82, 113
$0.6\ \delta_v$	0.20, 425	0.06, 476	0.46, 383	0.18, 499	0.70, 218
$0.8\ \delta_v$	0.18, 467	0.06, 492	0.42, 421	0.16, 499	0.68, 289
$1.0\ \delta_v$	0.16, 486	0.08, 499	0.36, 430	0.16, 498	0.56, 327
$2.0\ \delta_v$	0.08, 489	0.00, 499	0.00, -	0.00, -	0.10, 485

network with a total of 107 agents. In each run, agents are randomly distributed across nodes in the graph, with 30 agents set as non-cooperative agents randomly. The experiments are repeated 50 runs. Fig. 4 displays the results, which indicate: (1) The LSP maintains the highest success rate (consistently above 95%) across various non-cooperative behaviors and converges in the shortest time (convergence time steps always fewer than 75); (2) Introducing the Byzantine role function and inconsistent role function reduces the success rate and increases the convergence time steps for all approaches. Moreover, when the tampering function remains consistent, the inconsistent role function leads to a more significant decrease in success rate and a greater increase in convergence time steps than the Byzantine role function.

Number of non-cooperative agents & environmental noise. We investigated the impact of the number of non-cooperative agents $|N^{nc}|$ on the performance of various approaches. The communication topology was configured as a large-scale network, with the non-cooperative policy set to IRV, β_i set to 0.2, and ϕ set to 0.2. Table 2 displays the results. Each cell contains two metrics: the success rate and the average convergence time step, based on 50 trials. The observations are as follows: (1) The LSP approach consistently surpasses other approaches regardless of the number of non-cooperative agents. (2) Without non-cooperative behaviors, the QC, Linear, and LSP approaches all achieve the same success rates. This suggests that LSP's resilience mechanisms do not adversely affect the flocking control. (3) As non-cooperative agent count grows, the success rate of all approaches drops. However, the decline is more gradual for LSP. Similarly, the convergence time step increases for all approaches, but the rise is less pronounced for LSP. Under the same experimental settings, we studied the impact of environmental noise ω on various approaches. The results in Table 3 show that as the noise increases (from 0.2 times δ_v to 1.0 times δ_v), the performance of all approaches declines, except for the Linear approach which already performed poorly without noise. However, our approach still maintains a success rate of 56% even when the noise reaches 1.0 times δ_v. It's worth noting that, according to Definition 3, when the environmental noise exceeds δ_v, the velocity convergence condition Eq.(4) is almost impossible to ensure. This is also evident from the last row of Table 3, where all methods struggle to successfully achieve flock formation. In addition, the destructiveness of non-cooperation follows a bell-shaped curve as the Byzantine ratio β_i increases. The value $\beta_i = 0.2$ leads to the highest level of disruption.

7 DISCUSSION AND CONCLUSION

This study have presented a memory-based control approach for non-cooperation in multi-agent flocking, exploiting historical data from neighbors for detection and isolation of non-cooperative agents at long, short, and periodic oscillation scales. Experimental results highlight the effectiveness by its dynamic and robust defense, enhancing the resilience of multi-agent flocking control.

Finally, there are some areas in this paper that require further discussion. (1) The proposed approach considers convergence to a single velocity; future work may consider the more complex requirement of multiple-velocity convergence common in real-world scenarios. (2) As the application of flocking control extends to drones, autonomous underwater vehicles, and mobile robots, the necessity to anticipate and mitigate advanced attacks amplifies. We aim to consider defending against more complex non-cooperative behavior, such as undetectable attacks [47], game-theoretic non-cooperative agents [3], even though these threats have not yet been observed in current flocking control.

REFERENCES

[1] Yehia Abd Alrahman, Giuseppe Perelli, and Nir Piterman. 2020. Reconfigurable Interaction for MAS Modelling. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*. International Foundation for Autonomous Agents and Multiagent Systems, 7–15.

[2] Benjamin Aminof, Aniello Murano, Sasha Rubin, and Florian Zuleger. 2022. Verification of agent navigation in partially-known environments. *Artif. Intell.* 308 (2022), 103724.

[3] Ningbo An, Xiaochuan Zhao, Qishao Wang, and Qingyun Wang. 2023. Model-Free distributed optimal consensus control of nonlinear multi-agent systems: A graphical game approach. *J. Frankl. Inst.* 360, 12 (2023), 8753–8771.

[4] Ashay Aswale, Antonio López, Aukkawut Ammartayakun, and Carlo Pinciroli. 2022. Hacking the Colony: On the Disruptive Effect of Misleading Pheromone and How to Defend against It. In *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 27–34.

[5] Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony V. Robins. 2021. Pseudo-rehearsal: Achieving deep reinforcement learning without catastrophic forgetting. *Neurocomputing* 428 (2021), 291–307.

[6] John S. Baras and Xiangyang Liu. 2019. Trust is the Cure to Distributed Consensus with Adversaries. In *27th Mediterranean Conference on Control and Automation, MED 2019, Akko, Israel, July 1-4, 2019*. IEEE, 195–202.

[7] Chandreyee Bhowmick, Mudassir Shabbir, and Xenofon D. Koutsoukos. 2022. Attack-Resilient Multi-Agent Flocking Control Using Graph Neural Networks. In *30th Mediterranean Conference on Control and Automation, MED 2022, Vouliagmeni, Greece, June 28 - July 1, 2022*. IEEE, 300–305.

[8] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 119–129.

[9] Djamila Bouhata and Hamouma Moumen. 2022. Byzantine Fault Tolerance in Distributed Machine Learning : a Survey. *CoRR* abs/2205.02572 (2022).

[10] Andreas Brandstätter, Scott A. Smolka, Scott D. Stoller, Ashish Tiwari, and Radu Grosu. 2023. Multi-Agent Spatial Predictive Control with Application to Drone Flocking. In *IEEE International Conference on Robotics and Automation, ICRA 2023, London, UK, May 29 - June 2, 2023*. IEEE, 1221–1227.

[11] E Oran Brigham. 1988. *The fast Fourier transform and its applications*. Prentice-Hall, Inc.

[12] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. 2021. FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping. In *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*. The Internet Society.

[13] Chen Chen, Ge Chen, and Lei Guo. 2015. Consensus of flocks under M-nearest-neighbor rules. *J. Syst. Sci. Complex.* 28, 1 (2015), 1–15.

[14] Daniel Câmara. 2014. Cavalry to the rescue: Drones fleet to help rescuers operations over disasters scenarios. In *2014 IEEE Conference on Antenna Measurements & Applications (CAMA)*. 1–4. https://doi.org/10.1109/CAMA.2014.7003421

[15] Dimos V. Dimarogonas and Kostas J. Kyriakopoulos. 2007. On the Rendezvous Problem for Multiple Nonholonomic Agents. *IEEE Trans. Autom. Control.* 52, 5 (2007), 916–922.

[16] Katie Genter and Peter Stone. 2016. Adding Influencing Agents to a Flock. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*. ACM, 615–623.

[17] Katie Genter, Shun Zhang, and Peter Stone. 2015. Determining Placements of Influencing Agents in a Flock. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, Gerhard Weiss, Pinar Yolum, Rafael H. Bordini, and Edith Elkind (Eds.). ACM, 247–255.

[18] Wang Guanghua, Li Deyi, Gan Wenyan, and Jia Peng. 2013. Study on Formation Control of Multi-Robot Systems. In *2013 Third International Conference on Intelligent System Design and Engineering Applications*. 1335–1339.

[19] A. Haseltalab and M. Akar. 2015. Approximate byzantine consensus in faulty asynchronous networks. In *Proceedings of the 2015 American Control conference*. IEEE, Chicago, IL, USA, 1591–1596.

[20] Jian Hou, Zhiyong Chen, Mingyue Zhang, and Xiaomin Wang. 2022. Multi-armed bandit based distributed resilient consensus and its applications in social networks. *J. Frankl. Inst.* 359, 10 (2022), 4997–5013.

[21] Jian Hou, Fangyuan Wang, Lili Wang, and Zhiyong Chen. 2021. Reinforcement Learning Based Multi-Agent Resilient Control: From Deep Neural Networks to an Adaptive Law. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*.

[22] Jian Hou, Jing Wang, Mingyue Zhang, Zhi Jin, Chunlin Wei, and Zuohua Ding. 2023. Privacy-Preserving Resilient Consensus for Multi-Agent Systems in a General Topology Structure. *ACM Trans. Priv. Secur.* (mar 2023). https://doi.org/10.1145/3587933 Just Accepted.

[23] Xinyu Huang, Yunzhe Tian, Yifei He, Endong Tong, Wenjia Niu, Chenyang Li, Jiqiang Liu, and Liang Chang. 2020. Exposing Spoofing Attack on Flocking-Based Unmanned Aerial Vehicle Cluster: A Threat to Swarm Intelligence. *Secur. Commun. Networks* 2020 (2020), 8889122:1–8889122:15.

[24] H. J. LeBlanc and X. D. Koutsoukos. 2011. Consensus in networked multi-agent systems with adversaries. In *Proceedings of the 14th international conference on Hybrid systems: computation and control*. Chicago, IL, USA, 281–290.

[25] H. J. LeBlanc and X. D. Koutsoukos. 2012. Low complexity resilient consensus in networked multi-agent systems with adversaries. In *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control*. Beijing, China, 5–14.

[26] Haoyang Li, Qingqing Ye, Haibo Hu, Jin Li, Leixia Wang, Chengfang Fang, and Jie Shi. 2023. 3DFed: Adaptive and Extensible Framework for Covert Backdoor Attack in Federated Learning. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*. IEEE, 1893–1907.

[27] Jianan Li, Liang Li, and Shiyu Zhao. 2023. Predator–prey survival pressure is sufficient to evolve swarming behaviors. *New Journal of Physics* 25, 9 (2023), 092001.

[28] Yan Li, Hao Fang, and Jie Chen. 2015. Communication-based fault detection and isolation for multi-agent systems. In *10th Asian Control Conference, ASCC 2015, Kota Kinabalu, Malaysia, May 31 - June 3, 2015*. IEEE, 1–5.

[29] Guangliang Liu, Liang Zou, Qiuye Sun, Rui Wang, and Zhengqi Sui. 2023. Multiagent-Based Consensus Optimal Control for Microgrid With External Disturbances via Zero-Sum Game Strategy. *IEEE Transactions on Control of Network Systems* 10, 2 (2023), 659–670. https://doi.org/10.1109/TCNS.2022.3203896

[30] Xiangyang Liu and John S. Baras. 2014. Using trust in distributed consensus with adversaries in sensor and other networks. In *17th International Conference on Information Fusion, FUSION 2014, Salamanca, Spain, July 7-10, 2014*. IEEE, 1–7.

[31] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 6379–6390.

[32] Lifeng Ma, Zidong Wang, Qing-Long Han, and Yurong Liu. 2017. Consensus control of stochastic multi-agent systems: a survey. *Sci. China Inf. Sci.* 60, 12 (2017), 120201:1–120201:15.

[33] D. G. Mikulski, F. L. Lewis, E. Y. Gu, and G. R. Hudas. 2012. Trust method for multi-agent consensus. In *Conference on Unmanned Systems Technology XIV*. Baltimore, MD, USA, 1–15.

[34] Xiaowu Mu and Yuehua He. 2019. Leader-follower multi-agent flocking with bounded missing data and random communication radius. *Trans. Inst. Meas. Control* 41, 9 (2019), 2441–2450.

[35] Babak Nahid-Mobarakeh and Marcelo Godoy Simões. 2013. Multi-agent based fault detection and isolation in more electric aircraft. In *2013 IEEE Industry Applications Society Annual Meeting, Lake Buena Vista, FL, USA, October 6-11, 2013*. IEEE, 1–8.

[36] Martin Nowak and Karl Sigmund. 1993. A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game. *Nature* 364 (1993), 56–58.

[37] Reza Olfati-Saber. 2006. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on automatic control* 51, 3 (2006), 401–420.

[38] Sarah Perrin, Mathieu Laurière, Julien Pérolat, Matthieu Geist, Romuald Élie, and Olivier Pietquin. 2021. Mean Field Games Flock! The Reinforcement Learning Way. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*. ijcai.org, 356–362.

[39] Alyssa Pierson and Mac Schwager. 2015. Bio-inspired non-cooperative multi-robot herding. In *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*. IEEE, 1843–1849.

[40] G. Ramos, D. Silvestre, and C. Silvestre. 2020. General resilient consensus algorithms. *Internat. J. Control* (2020), 1–27.

[41] Craig W. Reynolds. 1987. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1987, Anaheim, California, USA, July 27-31, 1987*, Maureen C. Stone (Ed.). ACM, 25–34.

[42] Yara Rizk, Mariette Awad, and Edward W. Tunstel. 2019. Cooperative Heterogeneous Multi-Robot Systems: A Survey. *ACM Comput. Surv.* 52, 2 (2019), 29:1–29:31.

[43] Alejandro Ruiz-Esparza-Rodriguez, Moises S. Gonzalez-Chavez, and Victor J. Gonzalez-Villela. 2020. A Flocking Behavior Model with Artificial Potential Fields for the Coordinated Displacement of Robotic Swarms. In *2020 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*. 65–70. https://doi.org/10.1109/ICMEAE51770.2020.00019

[44] Kelsey Saulnier, David Saldana, Amanda Prorok, George J. Pappas, and Vijay Kumar. 2017. Resilient Flocking for Mobile Robot Teams. *IEEE Robotics Autom. Lett.* 2, 2 (2017), 1039–1046.

[45] Xiaofang Sun, Derrick Wing Kwan Ng, Zhiguo Ding, Yanqing Xu, and Zhangdui Zhong. 2019. Physical Layer Security in UAV Systems: Challenges and Opportunities. *IEEE Wirel. Commun.* 26, 5 (2019), 40–47.

[46] S. Sundaram and C. N. Hadjicostis. 2011. Distributed function calculation via linear iterative strategies in the presence of malicious agents. *IEEE Trans. Automat. Control* 56, 7 (2011), 1495–1508.

[47] Mahdi Taheri, Khashayar Khorasani, Iman Shames, and Nader Meskin. 2020. Undetectable Cyber Attacks on Communication Links in Multi-Agent Cyber-Physical Systems. In *59th IEEE Conference on Decision and Control, CDC 2020, Jeju Island, South Korea, December 14-18, 2020.* IEEE, 3764–3771.

[48] Guo-Xing Wen and C. L. Philip Chen. 2023. Optimized Backstepping Consensus Control Using Reinforcement Learning for a Class of Nonlinear Strict-Feedback-Dynamic Multi-Agent Systems. *IEEE Trans. Neural Networks Learn. Syst.* 34, 3 (2023), 1524–1536.

[49] Jian Xiao, Zhuoran Wang, Jinhui He, and Guohui Yuan. 2023. A graph neural network based deep reinforcement learning algorithm for multi-agent leader-follower flocking. *Inf. Sci.* 641 (2023), 119074. https://doi.org/10.1016/j.ins.2023.119074

[50] H. Zhang and S. Sundaram. 2012. Robustness of information diffusion algorithms to locally bounded adversaries. In *Proceedings of the 2012 American Control Conference.* Montreal, Quebec, Canada, 5855–5861.

[51] Mingyue Zhang, Zhi Jin, Jian Hou, and Renwei Luo. 2022. Resilient Mechanism Against Byzantine Failure for Distributed Deep Reinforcement Learning. In *IEEE 33rd International Symposium on Software Reliability Engineering, ISSRE 2022, Charlotte, NC, USA, October 31 - Nov. 3, 2022.* IEEE, 378–389.