Learning Verified Safe Neural Network Controllers for Multi-Agent Path Finding

Mingyue Zhang [1], Nianyu Li [2*], Yi Chen[1], Jialong Li [3], Xiao-Yi Zhang [4], Hengjun Zhao[1], Jiamou Liu [5], Wu Chen [1]

[1]College of Computer and Information Science, Southwest University, Chongqing, China
[2] Zhongguancun Laboratory (ZGC Lab), Beijing, China
[3]Department of Computer Science and Engineering, Waseda University, Tokyo, Japan
[4]School of Computer & Communication Engineering, University of Science and Technology Beijing, Beijing, China
[5]School of Computer Science, University of Auckland, Auckland, New Zealand
myzhangswu@swu.edu.cn, li_nianyu@pku.edu.cn, lijialong@fuji.waseda.jp, xiaoyi@ustb.edu.cn, chenwu@swu.edu.cn

Abstract

Multi-agent path finding (MAPF) is a safety-critical scenario where the goal is to secure collision-free trajectories from initial to desired locations. However, due to system complexity and uncertainty, integrating learning-based controllers with MAPF is challenging and cannot theoretically guarantee the safety of the learned controllers. In response, our study proposes a verified safe multi-agent neural control (VSMANC) approach for MAPF, focusing on the unified training of Decentralized Control Barrier Functions (DCBF) and controllers to enhence safety. VSMANC enables all agents to concurrently learn controllers and DCBFs using a unified loss function designed to maximize safety, adhere to standard control policies, and incorporate path-finding-related heuristics. We also propose a formal verification-guided retraining process to both verify the properties of the learned DCBFs and generate counterexamples for retraining, thereby providing a verified safety guarantee. We validate our approach through shape formation experiments and UAV simulations, demonstrating significant improvements in safety and effectiveness in complex multi-agent environments.

Introduction

Multi-agent path finding (MAPF) seeks collision-free trajectories for multiple agents moving from their initial to desired locations (Stern et al. 2019; Zhang et al. 2024; Liu and Zhang 2022). The importance of MAPF is clear in a wide range of applications: multi-agent monitoring (Ondrácek, Vanek, and Pechoucek 2015), cooperative delivery (Zong et al. 2022), and multi-robot patrolling (Joe and Lau 2023). Many methods have been developed to address the critical need for collision avoidance in these contexts (Borrmann et al. 2015; Cheng et al. 2020; Qin et al. 2021).

To address the collision avoidance problem in MAPF, there are three mainstream approaches: (1) model-based planning approaches (Chen et al. 2021; Zhou et al. 2019), which plan safe trajectories based on the system dynamics; (2) model-free methods based on safe multi-agent reinforcement learning (MARL) (Gu et al. 2023; Melcer, Amato, and Tripakis 2022), which impose safety constraints on MARL and learn a safe neural controller through interaction with the environment; (3) safe certificates-based methods (Zeng et al. 2023; Zhao et al. 2022), which use Lyapunov functions or barrier functions to constrain the controller's actions within a safe range. Among these, the third category provides the strictest theoretical safety guarantees due to its strong foundation in control theory and differential invariants (Liu, Zhan, and Zhao 2011).

In the line of safe certificates-based methods, Control Barrier Functions (CBFs) stand out due to their effectiveness in managing dynamic systems while ensuring critical safety constraints and maintaining boundary conditions (Cheng et al. 2019; Zeng et al. 2023; Zhao et al. 2021). CBFs are integrated into MAPF controllers to monitor system states and adjust control inputs, thus preventing agents from breaching safety boundaries. The application of CBFs in MAPF has led to significant innovations, from single-agent CBF-based shields designed to correct potentially hazardous actions (Cheng et al. 2019), to multi-agent CBF tailored for cooperative scenarios (Cai et al. 2021), decentralized CBF ensuring scalable collision-free operations (Wang, Ames, and Egerstedt 2017), and resilient CBF for response to the adversarial agents (Usevitch and Panagou 2023).

However, implementing CBFs in MAPF presents significant challenges, primarily due to the complexity and unpredictability inherent in dynamic multi-agent environments. Some existing work often relies on experts to manually design CBF using prior knowledge of system dynamics, a task that becomes particularly challenging under complex and uncertain conditions. Additionally, there is a growing interest in employing deep learning to train CBFs, especially suited for managing nonlinear and uncertain dynamics (Zeng et al. 2023; Zhao et al. 2021). However, since the learning method is data-driven, the rigorousness of the learned CBF resulting from zero training loss is not guaranteed. The inaccuracy of such learned CBFs can severely undermine the safety guarantees for the controller, thereby hindering the practical application of CBFs.

Our Works. In response, our study proposes a *verified safe multi-agent neural control* (VSMANC), which offers two key benefits: (1) it designs neural networks for CBFs and controllers, and jointly learns them from agent-environment interactions without relying on knowledge of the system dynamics; (2) it can formally analyze the input-output properties of the CBFs, strictly ensuring its rigorousness, thereby

*Corresponding Author.

enhancing the safety of the learned controller. Specifically, first, we introduce Decentralized Control Barrier Function (DCBF) that extends CBF to the multi-agent context, and construct a safe control set under the worst case, which is used to train the DCBF neural networks. Second, due to the mutual influence between the DCBF and the controller, we design a joint training framework. This framework uses a unified loss function specifically designed for MAPF tasks, considering both safety and efficiency of the path planning. Third, we propose a formal verification-guided retraining, which leverages formal analyzer of neural networks to verify the safety property of the DCBF neural networks, and retrains the DCBF neural networks under the counterexamples generated by the formal analyzer.

Finally, we conduct a set of comprehensive experiments on drones. Experimental results are indeed promising. The proposed method is highly scalable, applicable to various agent scales while maintaining very low collision rates. This has notably pushed the boundaries of MAPF and promoted the adoption of multi-agent safe learning.

Contributions. We make the following contributions:

- We introduce a supervised learning for jointly training DCBF and controller, underpinned by a novel unified loss function that enhances safety and efficiency.
- We develop a formal verification-guided retraining process to fine-tune the DCBF neural networks, thereby providing strict safety guarantees for the learned controller.
- We validate our methods through rigorous experiments on shape formation and simulations using DJI Mavic 3 Pro UAVs in outdoor scenarios, demonstrating the effectiveness and safety of our approach.

Related Work

Deep learning-based methods have been used in control of multi-agent systems for long, and recently regained popularity with the prosperity of multi-agent reinforcement learning equipped with deep neural networks (DNN) (Lowe et al. 2017; Zhang et al. 2021). The applications (such as MAPF) of them are typically safety-critical and potentially malicious (Amir, Sharon, and Stern 2015), thus, the safety of learning-based control is attracting increasing attention.

Researchers are beginning to introduce Control Barrier Functions (CBFs) into the learning-based control. CBFs, effective tools from control theory and formal methods, serve as proofs for the satisfaction of the certain safety constraints or boundary conditions of a system (Cheng et al. 2019; Zhao et al. 2021; Zeng et al. 2023). (Chen, Peng, and Grizzle 2018) proposed a supervisory control algorithm based CBF for low-speed vehicles, which relies on specific problem structures. Some studies focused on learning CBF from data, i.e., using DNN to approximate CBF and using the system's running data to train the DNN (Boffi et al. 2020; Zhao et al. 2021). Some studies dealt with uncertainties in the system, including introducing martingale theory into CBF-based control (Black et al. 2023), incorporating Gaussian processes into CBFs (Khan, Ibuki, and Chatterjee 2021), and combining Bayesian neural networks with CBFs (Zeng et al. 2023), among others. For guaranteeing safety in multi-agent systems, existing studies extend the single-agent CBF into multi-agent CBF. (Borrmann et al. 2015) designed a CBF for swarm with perfectly known system dynamics. (Wang, Ames, and Egerstedt 2017) assumed a worst-case uncertainty bounds, and modified the controllers to formally satisfy safety constraints for provably collision-free behaviors in multi-robot systems. (Chen et al. 2021) proposed a scalable decentralized controller under the CBF framework, but its reliance on online integration of dynamics can be computationally challenging for complex systems. (Cheng et al. 2020) addressed scenarios with unknown system dynamics and adopted matrix-variate Gaussian process for learning the robust multi-agent CBFs. (Qin et al. 2021) introduced a joint training approach that effectively learns control policies and multi-agent CBFs concurrently, showcasing excellent scalability for a large number of agents.

Additionally, formal analysis and verification techniques for the safety and robustness of DNNs have been proposed. For example, (Jin et al. 2020) developed a PCA-based framework to analyze DNNs generalization errors. (Lin et al. 2019) constructed a nonlinear optimization problem to assess the robustness of DNNs using sigmoid activation functions. Marabou is one of the most widely used tools for formal analysis of DNN's properties, supporting various types of networks and nonlinear activation functions (Katz et al. 2019). (Athavale et al. 2024) builds on Marabou by combining self-composition to utilize existing reachability analysis techniques and introducing a novel abstraction of the softmax function, suitable for automated verification. To the best of our knowledge, our work is the first approach to incorporate formal verification into learning-based CBF methods and the first to use verified CBFs to ensure controller safety.

Preliminaries and Problem Statement

This section presents the multi-agent dynamics, and introduces the safe learning problem in MAPF. Throughout the paper, $\mathbb{R}$ is the set of real numbers, and $\|\cdot\|_2$ is the 2-norm.

Multi-Agent Dynamics

The multi-agent system in MAPF consists of an operating physical environment and multiple mobile agents.

Definition 1. *The environment model is defined as a 5-tuple $\mathcal{E} = \langle N, S, S_0, S_g, \mathcal{O}\rangle$, where: (1) N is a finite set of n agents labeled by $1, ..., n$; (2) $S \subseteq \mathbb{R}^m$ is the location space of the environment where the agents exist, m is the dimensionality of the locations, and $\vec{s}(t) = [s_1(t), ..., s_n(t)]$ is the joint location of agents at time t where $s_i(t) \in S$ is agent i's location at time t; (3) $S_0 = \bigcup_{i=1}^{n} S_{i,0}$ is a set of initial locations for all agents, where $S_{i,0} \subseteq S$ is the initial location set of agent i; (4) $S_g = \bigcup_{i=1}^{n} S_{i,g}$ is a set of goal locations for all agents, where $S_{i,g} \subseteq S$ is the goal location set of agent i; (5) $\mathcal{O} \subseteq S$ is the location space of obstacles, it satisfies that $\mathcal{O} \cap S_g = \emptyset$ and $\mathcal{O} \cap S_0 = \emptyset$.* □

In line with other solutions (Dimarogonas and Kyriakopoulos 2007; Hou et al. 2021; Zeng et al. 2023), the dynamics for each agent is formalized as follows.

Definition 2. *Given an environment $\mathcal{E}$, the agent dynamics is defined as $\mathcal{A} = \langle V, A, \phi, f\rangle$, where: (1) $V = \{V_1, ..., V_n\}$*

is the velocity space, and $v_i(t) \in V_i$ denotes the velocity of agent i at time t; (2) $A = \{A_1, ..., A_n\}$ is the action space, and $a_i(t) \in A_i$ denotes the acceleration of agent i at time t; (3) $\phi \in \Phi$ represents bounded noise induced by environment, where $\Phi \subseteq \mathbb{R}^m$ is the range of noise; (4) $f = \{f_1, ..., f_n\}$ is a set of dynamics functions for all agents, where agent i's dynamics is $\dot{s}_i(t) = f_i(s_i(t), a_i(t))$. □

f_i is considered in a discretization form, where continuous time is divided into discrete time steps. Thus, the location dynamics is: $s_i(t+1) = s_i(t) + v_i(t)\Delta t$. The velocity dynamics is: $v_i(t+1) = v_i(t) + a_i(t)\Delta t + \phi_i(t)$, where Δt is a tiny increment of time. $s_i(t+1)/v_i(t+1)$ is the location/velocity of agent i at time step $t+1$, and $\phi_i(t)$ is the environmental noise for agent i at time step t. In line with other solutions (Qin et al. 2021; Zhao et al. 2021; Zeng et al. 2023), we assume that f_i is unknown. Each agent only observes its own location and those of its neighbors. Let $\mathcal{N}_i(t) \subseteq N$ be the set of agent i's neighbors at time t. The observation $o_i(t) \in \mathbb{R}^{m \times |\mathcal{N}_i(t)|}$ is the joint locations of i's neighbors. Ω_i is i's observation space. The acceleration of each agent is controlled by the controller $\pi_i : S^{1+|\mathcal{N}_i(t)|} \to A_i$. $\pi_i(s_i(t), o_i(t)) = a_i(t)$ denotes that when agent i is at state $s_i(t)$ and observes $o_i(t)$, it will take action $a_i(t)$.

Problem Statement

The safety constraint requires that all agents should avoid collisions with each other and collisions to the obstacles while moving from their initial to desired locations.

Definition 3 (Safety)**.** *For an environment $\mathcal{E}$ and an agent dynamics $\mathcal{A}$, the system consisting of all agents in N is safe if: (1) $\forall i \in N$ there does not exist $t \geq 0$ s.t. $s_i(t) \in \mathcal{O}$; (2) for any two agents $i, j \in N$ there does not exist $t \geq 0$ s.t. $\|s_i(t) - s_j(t)\|_2 \leq \mathcal{K}$, where $\mathcal{K} \in \mathbb{R}^+$ is a safe distance.* □

Problem 1. *Given a set of agents N, a set of initial locations S_0, a set of goal locations S_g, the controller for each agent in MAPF is defined as a function $\pi_i : S^{1+|\mathcal{N}_i(t)|} \to A_i$. Each controller should find a trajectory $[s_i(0), ..., s_i(t)]$ from initial to goal location, and **minimize time steps** as much as possible while **meeting safety constraints**.* □

Since the agent dynamics functions are unknown, we parameterize agent i's controller as a neural network $\pi_i^{\omega_i}$, where ω_i denotes the weights of the neural network, and use the data generated by the environment to train the network.

Decentralized Control Barrier Function

The concept of CBF plays an important role in safety verification of dynamics systems. To ensure the controllers for multiple agents adhere to the safety constraints, *Decentralized Control Barrier Function* (DCBF) was introduced (Qin et al. 2021), which extends the CBF to multi-agent systems.

Theorem 1. *Given an environment $\mathcal{E}$ and an agent dynamics $\mathcal{A}$, for each agent $i \in N$, the multi-agent dynamic system is safe, if there exists a continuously differentiable function $h_i : S \times \Omega_i \to \mathbb{R}$ s.t. $\big(\forall (s_i, o_i) \in S_{i,0} \times \Omega_i . h_i(s_i, o_i) \geq 0\big) \bigwedge \big(\forall (s_i, o_i) \in S_{i,d} \times \Omega_i . h_i(s_i, o_i) < 0\big) \bigwedge \big(\forall (s_i, o_i) \in \{(s_i, o_i) | h_i(s_i, o_i) \geq 0\} . \mathcal{L}_{f_i} h_i(s_i, o_i) + \lambda h_i(s_i, o_i) > 0\big)$, where $\mathcal{L}_{f_i} h_i(s_i, o_i)$ is the* Lie derivative *of $h_i(s_i, o_i)$ along $f_i(s, a)$ (i.e., the inner product of f_i and the gradient of h_i), $S_{i,d}$ is the agent i's dangerous set, which is designed based on the safety constraints, and $\lambda > 0$ is a coefficient. Such h_i is the decentralized control barrier function for agent i.* □

Verified Safe Multi-Agent Neural Control

The intuition behind our proposed learning method is that integrating DCBF neural networks, verified through formal verification, into the learning process of the neural network controllers for ensuring the safety. As illustrated by the framework in Fig.1, besides an environment (i.e., multi-agent dynamic system) for generating the training data, the learning framework for verified safe multi-agent controller consists of two parts: (1) *Supervised learning of DCBF and controller*, which parameterizes the DCBFs for safety constraints and the controllers for path planning as neural networks, leverages the DCBF neural networks to constrain the output of the controller neural networks within a safe range, and jointly trains them due to the coupling relationship between the DCBF and controller. (2) *Formal Verification-Guided Retraining*, which designs a formal verification-based process to verify DCBF neural networks satisfies safety constraints, stopping the whole learning process if constraints are satisfied or generating counterexamples to retrain the DCBF neural networks if they are not.

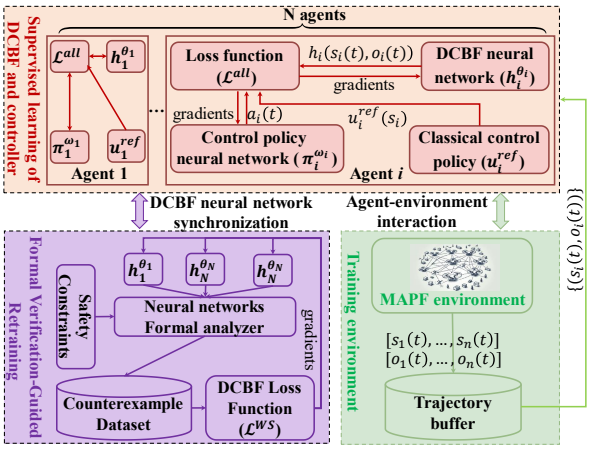


Figure 1: Our framework

Supervised Learning of DCBF and Controller

To achieve the learning objective, we construct a safe control set for DCBFs under the worst-case scenario, design neural networks for both the DCBFs and controllers, and create a loss function for training these neural networks.

Safe Control Set under the Worst Case. For simplicity of analysis, we assume that at each time step, each agent has a probability β of taking the worst possible action, i.e., moving towards a collision with the nearest agent. Let Δt be the time interval and V_{max} be the maximum velocity. Next, we investigate the *worst-case safe distance*, denoted as $\mathcal{K}_B > \mathcal{K}$, which is the minimum required to ensure that the agent does not collide in the worst case. In the worst

case, i.e., agent i and j are heading towards each other, it takes at least $\frac{\mathcal{K}_B-\mathcal{K}}{2V_{max}}$ time for them to violate the safety constraint (i.e., $\|s_i(t_1)-s_j(t_1)\|_2 > \mathcal{K}$). In terms of time steps, it requires a minimum of $\frac{\mathcal{K}_B-\mathcal{K}}{2V_{max}\Delta t}$ steps. Hence, the probability of two agents violating the safety constraint within a distance of $\mathcal{K}_B$ is at most $\beta^{\frac{\mathcal{K}_B-\mathcal{K}}{V_{max}\Delta t}}$. Given a safety threshold P_S (i.e., the probability of an agent colliding at each time step is less than P_S), $\mathcal{K}_B$ must satisfy:

$$\beta^{\frac{\mathcal{K}_B-\mathcal{K}}{V_{max}\Delta t}} \leq 1-P_s \tag{1}$$

Based on Eq.(1), we can derive the minimum value of $\mathcal{K}_B$.

$$\begin{aligned} &\lg^{\beta^{\frac{\mathcal{K}_B-\mathcal{K}}{V_{max}\Delta t}}} \leq \lg^{1-P_s} \Rightarrow \frac{\mathcal{K}_B-\mathcal{K}}{V_{max}\Delta t}\lg^{\beta} \leq \lg^{1-P_s} \\ &\Rightarrow \mathcal{K}_B-\mathcal{K} \geq V_{max}\Delta t\frac{\lg^{1-P_s}}{\lg^{\beta}} \\ &\Rightarrow \mathcal{K}_B \geq V_{max}\Delta t \lg_{\beta}^{1-\alpha(P_s)}+\mathcal{K} \end{aligned} \tag{2}$$

where $\lg = \log_{10}$, $\beta < 1$, $\lg^{\beta} < 0$. Let $\mathcal{K}_{BS} = V_{max}\Delta t \lg_{\beta}^{1-P_s}$. The safe constraint for two agents is rewritten as follows: for any two agents $i, j \in N$, there does not exist $t \in [0, T_{max}]$ s.t. $\|s_i(t)-s_j(t)\|_2 \leq \mathcal{K}_{BS}+\mathcal{K}$.

In the same way, we get the safe constraint to avoid collisions with obstacles under the worst case: for all $i \in N$ there does not exist $t \in [0, T_{max}]$ s.t. $\text{dis}(s_i(t), \mathcal{O}) \leq \mathcal{K}_{BS}$, where $\text{dis}(s_i(t), \mathcal{O}) = \min_{s\in\mathcal{O}} \|s_i(t)-s\|_2$.

The *worst-case safe control set* is designed as follows:

$$\begin{aligned} S_s = \{(s_1,...,s_n) \in S^n | \forall i \in [1,n].\text{dis}(s_i,\mathcal{O}) > \\ \mathcal{K}_{BS} \bigwedge \forall i,j \in [1,n]. \|s_i-s_j\|_2 > \mathcal{K}_{BS}+\mathcal{K}\} \end{aligned} \tag{3}$$

Leveraging the DCBF proposition (Qin et al. 2021) and Eq.(3), the *dangerous set* of agent i is defined as follows:

$$\begin{aligned} S_{i,d} = \{s_i \in S | \text{dis}(s_i,\mathcal{O}) \leq \mathcal{K}_{BS} \\ \bigvee \exists j \in [1,n]. \|s_i-s_j\|_2 \leq \mathcal{K}_{BS}+\mathcal{K}\} \end{aligned} \tag{4}$$

The Unified Loss Function. Based on Eq.(4), we design a unified loss function to jointly train the DCBF neural networks $\pi_i^{\omega_i}$ and the neural network controllers $h_i^{\theta_i}$.

Computing the DCBF (refer to Theorem 1) requires the Lie derivative $\mathcal{L}_{f_i}h_i(s_i,o_i)$, which is computed as follows:

$$\begin{aligned} \mathcal{L}_{f_s}h_i(s_i,o_i) &= (\nabla h_i)\cdot f(s_i,\pi_i(s_i,o_i)) \\ &= \nabla_{s_i}h_i \cdot f_i(s_i,\pi_i(s_i,o_i)) + \nabla_{o_i}h_i \cdot f_{o,i} \end{aligned} \tag{5}$$

where $f_{o,i}(\vec{s},\vec{v}) = o_i$ is the time derivative of the observation, which depends on the behavior of other agents.

Given the observed dependency of the Lie derivative on both h_i and π_i, we jointly train $h_i^{\theta_i}$ and $\pi_i^{\omega_i}$. The loss function of the two neural networks consists of three parts. The first part is the *worst-case safe loss*:

$$\begin{aligned} &\mathcal{L}_i^{WS} = c_1 \sum_{s_i\in I_i} \text{ReLU}(\gamma_1 - h_i^{\theta_i}(s_i,o_i)) + \\ &c_2 \sum_{s_i\in S_{i,d}} \text{ReLU}(\gamma_2 + h_i^{\theta_i}(s_i,o_i)) + \\ &c_3 \sum_{s_i\in S_{i,h}} \text{ReLU}(\gamma_3 - \mathcal{L}_{f_i}h_i^{\theta_i}(s_i,o_i) - \lambda h_i^{\theta_i}(s_i,o_i)) \end{aligned} \tag{6}$$

where $S_{i,h} = \{(s_i,o_i) | h_i^{\theta_i}(s_i,o_i) \geq 0\}$, and c_1, c_2, c_3 are positive coefficients. γ_1, γ_2, γ_3 are three small positive tolerances, the role of which is to increase the generalization of $h_i^{\theta_i}$ and $\pi_i^{\omega_i}$, i.e., to enforce zero loss on the non-sampled (s_i,o_i). For the Lie derivative, we employ a discretization approach to approximately compute it: $\mathcal{L}_{f_i}(h_i(s_i,o_i)) = \frac{1}{\Delta t}\big(h_i(s_i(t+\Delta t), o_i(t+\Delta t)) - h_i(s_i(t),o_i(t))\big)$.

In addition to satisfying safety constraints, another objective of the controller is to plan a trajectory from the initial location to the goal location. We introduce a classical control approach to guide the controller to the goal. Hence, the second part is the *classical control loss*:

$$\mathcal{L}_i^{ref} = c_4 \sum_{s_i\in S} \left\| \pi_i^{\omega_i}(s_i,o_i) - u_i^{ref}(s_i) \right\|_2 \tag{7}$$

where $u_i^{ref}(s_i)$ is a classical control policy/approach (e.g., PID, LQR or MPC), and $c_4 > 0$ is a weight coefficients.

To make the agent reach the goal as soon as possible, the third part is the *goal distance loss*:

$$\mathcal{L}_i^{goal} = c_5 \sum_{s_i\in S} \left\| \frac{\pi_i^{\omega_i}(s_i,o_i)}{\|\pi_i^{\omega_i}(s_i,o_i)\|_2} - \frac{g_i-s_i}{\|g_i-s_i\|_2} \right\|_2 \tag{8}$$

where $g_i \in S_{i,g}$ is the current goal location of agent i.

c_1, c_2, c_3, c_4, c_5 are set to 1, 1, 1, 0.1, and 0.05, respectively. $\gamma_1 = \gamma_2 = \gamma_3 = 0.01$. The comprehensive loss function of the neural networks for all agents is:

$$\mathcal{L}^{all} = \sum_{i\in N}\mathcal{L}_i^{WS} + \sum_{i\in N}\mathcal{L}_i^{ref} + \sum_{i\in N}\mathcal{L}_i^{goal} \tag{9}$$

Formal Verification-Guided Retraining

Theoretically, a DCBF that fully satisfies Theorem 1 can provide strict safety guarantees for the controller's output. However, since the approximation of DCBF with neural network is data-driven, the rigorousness of $h_i^{\theta_i}$ resulted from 0 training loss is not guaranteed. Therefore, we enhance the accuracy of $h_i^{\theta_i}$ through formal verification of the neural network, thereby improving safety of the controller.

Due to the limitations of existing neural network formal analyzer, some special computations/operations in certain neural networks cannot be directly verified, and the time cost for verifying large-scale neural networks is very high. To preform the verification, we distill $h_i^{\theta_i}$ using soft target knowledge distillation (Hinton, Vinyals, and Dean 2015). $h_i^{\theta_i}$ serves as the teacher network, and a Multilayer Perceptron (with ReLU activation functions for each layer) as the student network, denoted as $h_{stu,i}^{\theta_i'}$. Extract (s_i,o_i) data from the episode buffer as features and use the corresponding outputs of $h_i^{\theta_i}$ as labels to train $h_{stu,i}^{\theta_i'}$. The loss function for $h_{stu,i}^{\theta_i'}$ is Mean Squared Error (MSE):

$$\mathcal{L}_i^{h\_student} = \frac{1}{m}\sum_{t=1}^{m}\left(h_i^{\theta_i}(s^t,o^t) - h_{stu,i}^{\theta_i'}(s^t,o^t)\right)^2 \tag{10}$$

where (s^t,o^t) refers to the t-th state and observation in the training data, m is the number of training data points.

We verify $h_{stu,i}^{\theta_i'}$ using the Marabou (Katz et al. 2019) under the following properties: (1) $\forall(s,o) \in S_{i,0} \times$

$\Omega_i.h^{\theta'_i}_{stu,i}(s,o) \geq 0$; (2) $\forall(s,o) \in S_{i,d} \times \Omega_i.h^{\theta'_i}_{stu,i}(s,o) < 0$; (3) $\forall(s,o) \in S_s \times \Omega_i.h^{\theta'_i}_{stu,i}(s,o) \geq 0$. If $h^{\theta'_i}_{stu,i}$ fails to satisfy these three properties, Marabou returns a set of counterexamples $\{(s,o)\}$. These counterexamples are used to retrain $h^{\theta_i}_i$, and the loss function for retraining $h^{\theta_i}_i$ is MSE:

$$\mathcal{L}^{h\_ret}_i = \frac{1}{m}\sum_{t=1}^{m}\left(h^{\theta_i}_i(s^t,o^t) - y^t\right)^2 \tag{11}$$

where (s^t, o^t) refers to the t-th counterexample, m is the number of counterexamples, y^t is computed as follows: if $(s^t,o^t) \in S_{i,0} \times \Omega_i \cup S_s \times \Omega_i$, $y^t = \mathcal{C}$; otherwise, $y^t = -\mathcal{C}$, where $\mathcal{C} > 0$ is a real number.

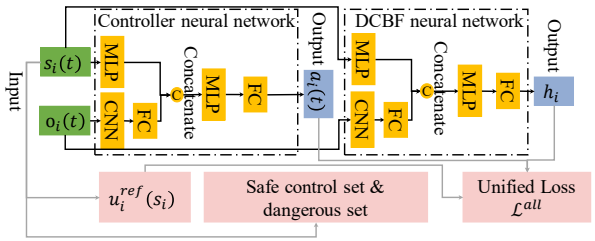


Figure 2: The loss function and neural network architecture.

Implementation

Fig. 2 presents the neural network architecture of the controller and the DCBF, which consists of Multilayer Perceptron (MLP), Convolutional Neural Network (CNN) and fully connected layer (FC). The controller for each agent is homogeneous. Regardless of the agent, given the same location and observation, the action taken should be identical. Therefore, all agents share a neural network controller π^ω and a DCBF neural network h^θ. Due to the dynamic nature of an agent's neighbors, using neighbors' locations directly as observations would result in a variable input length of π^ω. Inspired by grid-wise control (Han et al. 2019), we design a masking representation as follow. The observation o_i for agent i is constructed by centering a $r \times r$ square grid (where r is the field of view's radius) around agent i. This grid is transformed into the matrix $o_{k\times k}$. A grid value at $o[i,j]$ is designated as 1 if it is occupied by an agent or obstacle; otherwise, it's assigned a value of 0.

Experiments

This section delineates the evaluations and addresses the following questions: (1) the performance (including safety, efficiency and generalization capability) of different MAPF solutions; (2) the ablation study of our method; (3) the practicality the proposed method in the actual quadcopter drones.

Experimental Setup

Shape formation task. For the performance comparison, we have chosen the UAV shape formation task, a classic task in the MAPF, as our experimental benchmark. In this task, each drone has a unique starting location and a designated goal location. Together, the goal locations of all drones create a distinct pattern. We have established 200 patterns, which include English letters, combinations of multiple letters, various geometric shapes, and so on.

Obstacle Setting. As shown in Fig.4, we have chosen three types of scenarios for simulation evaluation: (1) a random map, characterized by cuboidal obstacles that are positioned arbitrarily; (2) a cross map, wherein the spatial arrangement of obstacles emulates a crossroads; (3) a street map, abstracted from the open building dataset pertaining to Portland, USA (Burian et al. 2002).

Comparison Approaches and Hardware. Four compared approaches are considered, namely (1) MDBC (Qin et al. 2021), which learns a safe controller within the constraints of DCBF; (2) S2M2 (Chen et al. 2021), a scalable and safe MAPF solution that leverages mixed integer-linear program; (3) MADDPG-safe, which incorporates the safety reward to the reward function of MADDPG, a pioneering work on multi-agent reinforcement learning (Lowe et al. 2017); (4) ALF (Chu et al. 2023), an MAPF solution rooted in artificial potential fields, optimally suited for scenarios with minimal to no obstacles; (5) CS-MARL, a state-of-the-art safe MARL used in autonomous driving (Zheng and Gu 2024). All the simulation experiments are conducted on a desktop running Ubuntu 16, powered by Intel(R) Core(TM) i7-7700 CPU@3.6GHz and an Nvidia Quadro P600 GPU.

Evaluation Criteria. For evaluating the safety, we define a *safety rate* (Qin et al. 2021): $\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{t\in T}[\mathbb{I}(s_i(t) \notin S_{i,d})]$ where $\mathbb{I}(a)$ is the indicator function that is 1 when a is true or 0 otherwise, T is the termial time step, n is the number of agents. For evaluating the efficiency, we use the *running time step* from the initial takeoff of the first drone to the moment the last drone reaches its goal location. In our experiment, each time step simulates 0.1 seconds ($\Delta t = 0.1s$).

Results and Analysis

Performance. Fig.3 presents a comparative performance analysis of the proposed VSMANC method against the aforementioned five methods, focusing on safety rate and running time steps across the three testing scenarios and varying drone scales. The experimental results demonstrate that: (1) Irrespective of variations in drone scale, our method consistently outperforms the competing methods in both safety rate and the efficiency (i.e., running time steps); (2) Relative to the second best-performing method, our method enhances the safety rate by margins ranging from 3% to 37% and reduces the execution time steps by 16% to 30%. (3) As the scale of drones increases, the safety rate of all methods decreases, while the time steps increase. Compared to other methods, the decrease in safety rate and the increase in time steps are relatively smaller for our approach. In addition, it is important to note that MADDPG-safe and CS-MARL struggle to converge in large-scale scenarios with more than 32 drones. Therefore, for comparison, we use the policy networks trained with 32 drones to test with 64 drones.

We further examine the generalization capability of our proposed method. Generalization capability refers to the method's performance in terms of safety rate and running time steps when there are differences between the MAPF environment configurations during training and testing. As shown in Fig.5, we set the number of drones to 8 and use a random map during training. In the testing phase, the number of drones is expanded from 4 to 1024, and the maps in-

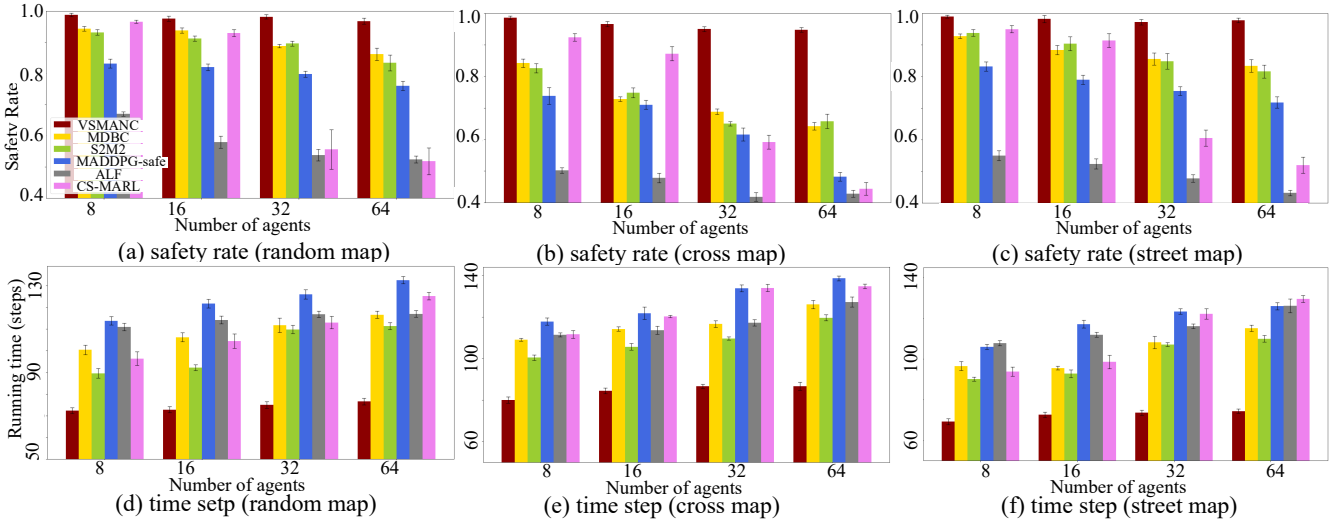


Figure 3: Safety rate and running time step on different scenarios. The results are taken after each method converged and are averaged over 20 independent trials.

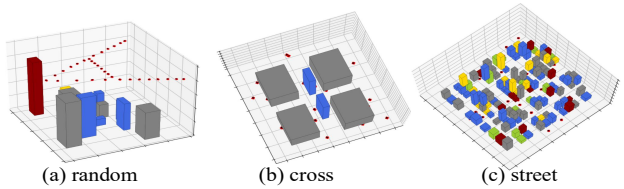


Figure 4: Three types of MAPF scenarios.

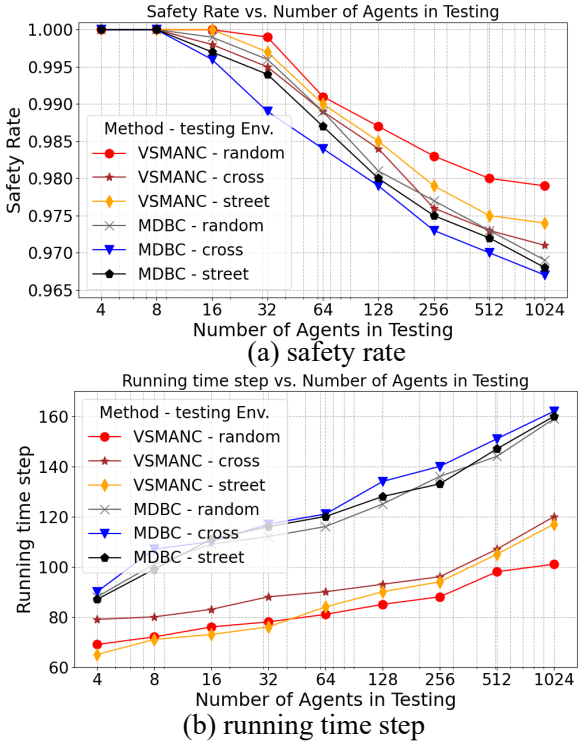


Figure 5: Generalization of VSMANC and MDBC.

clude random map, cross map, and street map. Since other methods, except for MDBC, struggle to scale beyond 64 drones and do not address generalization capability, we only compare our method with MDBC. The experimental results indicate that: (1) Under the same testing conditions, our method outperforms MDBC in both safety rate and running time steps; (2) Our method has a particular advantage in running time steps, exceeding MDBC by up to 36.5%.

Ablation Study. The key components affecting the performance of VSMANC include the three parts of the unified loss (i.e., $\mathcal{L}^{WS}$,$\mathcal{L}^{ref}$, and $\mathcal{L}^{goal}$) and the formal verification-guided retraining. In the ablation study, we examine the impact of each of these four components individually. We design four variants of VSMANC (i.e., V(w/o)$\mathcal{L}^{WS}$,, V(w/o)$\mathcal{L}^{ref}$, V(w/o)$\mathcal{L}^{goal}$, V(w/o)VS), corresponding to VSMANC without $\mathcal{L}^{WS}$ loss, without $\mathcal{L}^{ref}$ loss, without $\mathcal{L}^{goal}$ loss, and without formal verification-guided retraining, respectively. The results in Tab.1 show that: (1) $\mathcal{L}^{WS}$ loss and formal verification-guided retraining have a significant impact on the safety rate; removing them leads to a substantial decrease in safety rate; (2) $\mathcal{L}^{ref}$ loss and $\mathcal{L}^{goal}$ loss significantly affect the running time step, and $\mathcal{L}^{ref}$ loss also slightly influences the safety rate; (3) Comparing VSMANC, V(w/o)$\mathcal{L}^{WS}$, and V(w/o)VS shows that the formal verification-guided retraining greatly enhances the controller's safety, and using the retraining alone without $\mathcal{L}^{WS}$ loss can ensure a certain level of safety.

map	VSMANC	V(w/o)$\mathcal{L}^{WS}$	V(w/o)$\mathcal{L}^{ref}$	V(w/o)$\mathcal{L}^{goal}$	V(w/o)VS
random	0.99, 75	0.89, 79	0.97, 101	0.99, 119	0.90, 76
cross	0.99, 87	0.87, 89	0.98, 113	0.98, 124	0.90, 85
street	0.99, 74	0.88, 75	0.97, 98	0.99, 117	0.91, 72

Table 1: Ablation Study ($|N| = 32$, the results are in the form of "safety rate, running time step".)

The formal verification-guided retraining not only verifies safety but also generates counterexamples for train-

ing the DCBF. We also compare the effectiveness of generating counterexamples using this method to that of random sampling (i.e., randomly selecting input states from the input space, where any input state that causes the DCBF to violate safety constraints is used as a counterexample). In Tab.2, $\mathcal{C}$ is the counterexample label value mentioned in Eq.(11). A round involves generating counterexamples using formal verification or random sampling, taking the DCBF and safety constraints as inputs, followed by retraining the DCBF using the generated counterexamples. Multiple rounds mean repeating this process several times. The results in Tab. 2 show that: (1) As $\mathcal{C}$ increases, the safety rate improves for both formal verification-based method and the random sampling; as the number of rounds increases, the safety rate also improves for both methods; (2) Under the same $\mathcal{C}$ and number of rounds, the formal verification-based method achieves a higher safety rate than the random sampling, demonstrating its advantage in retraining the DCBF.

	Formal verification-based				Random			
Round	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=3$	$\mathcal{C}=4$	$\mathcal{C}=1$	$\mathcal{C}=2$	$\mathcal{C}=3$	$\mathcal{C}=4$
#1	0.954	0.970	0.991	0.994	0.947	0.949	0.951	0.956
#2	0.971	0.975	0.993	0.997	0.950	0.951	0.955	0.959
#3	0.983	0.984	0.993	0.997	0.954	0.956	0.960	0.961
#4	0.988	0.995	0.994	0.998	0.962	0.962	0.965	0.967
#5	0.995	0.997	0.997	0.998	0.962	0.964	0.967	0.968

Table 2: The impact of the formal verification-based method and random sampling on DCBF retraining performance.

Tab.3 presents the results of distilling the DCBF network using different student networks architectures. "FC $\times n$" indicates that the student network has n FC layers, and "$k_1+...+k_n$" represent the number of neurons in each layer. We measure the similarity between the student network and the original network: by randomly selecting 100 states from the input space, we record whether the safety constraints are violated in both the student and original networks, thereby obtaining the true positives (TP, cases where both the student and original networks violate safety constraints), false positives, true negatives, and false negatives, and then calculate the F1 score. The results in Tab.3 reveal that: (1) Increasing the complexity of the student network does not necessarily improve the F1 score; the best F1 score is achieved when the student network is set to FC $\times 2$ (64+32). (2) The F1 score and safety rate are positively correlated; a higher F1 score leads to a higher safety rate of the controller.

FC×1	FC×1	FC×2	FC×2	FC×2	FC×2
64	128	$16+8$	$32+16$	$64+32$	$128+64$
0.25, 0.94	0.22, 0.92	0.33,0.95	0.54,0.97	0.69,0.99	0.31,0.94

FC×3	FC×3	FC×4	FC×4
$32+16+8$	$64+128+64$	$32+16+8+4$	$32+32+8+8$
0.59, 0.97	0.22, 0.92	0.30,0.94	0.29,0.94

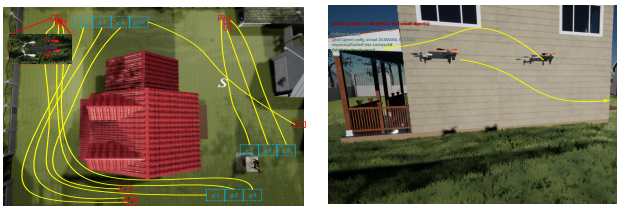
FC×4	FC×5	FC×5
$32+16+8+4$	$32+16+8+4+2$	$128+64+32+32+8$
0.37, 0.95	0.51,0.96	0.41,0.96

Table 3: The performance of the DCBF neural network after distillation using different architectures (The results are in the form of "F1 score, safety rate").

Approach	VSMANC	MDBC	S2M2	MADDPG-safe	ALF	CS-MARL
Proportion	1.00	0.90	0.85	0.70	0.45	0.80

Table 4: Proportion of safe trajectories in the outdoor simulation experiment, calculated as the number of collision-free episodes divided by the total number of episodes (20 trials).

Practicality. Fig.6 presents the outdoor simulation, where the blue rectangles are the goal locations for the drones, the red rectangles signify the drones themselves, and the yellow lines trace their flight paths generated by VSMANC. Fig.6(b) depicts the evasive maneuvers undertaken by two drones upon encountering each other at location s. In the experiment, we set the condition that once a collision occurs, the episode is terminated. The results in Tab.4 show that VSMANC demonstrates very high safety in real-world scenarios, with no collisions occurring in any of the 20 trials, whereas other methods experienced multiple collisions.



(a) Top view of the outdoor simulation (b) The case of two drones avoiding each other

Figure 6: Outdoor simulation ($|N| = 10$).

Limitations and Future Work

Aiming at the safety concerns in the Multi-Agent Path Finding (MAPF), we propose a verified safe multi-agent neural control approach that augments the Decentralized Control Barrier Functions (DCBFs) and the formal verification to guarantee the safety of the agents' paths, and design a unified loss function for concurrently training both the DCBFs and controllers. The performance and practicality of the proposed method are validated through simulation experiments.

Despite these advancements, there are some limitations in the proposed method. (1) Our learning framework is based on supervised learning, requiring standard control policies and heuristic path-finding rules for guidance. (2) The formal verification-guided method uses distillation to enable DCBF neural network verification, but this may result in some loss of rigorousness, and could potentially weaken the safety guarantees for the controller. (3) The UAV simulations employ double-integrator dynamics, a basic model that may not fully capture the potential of our learning approach in more sophisticated scenarios. To address these limitations, in future work, we plan to: (1) incorporate multi-agent reinforcement learning to refine learning capabilities; (2) introduce theoretically grounded abstraction or approximation techniques from formal methods to simplify the DCBF neural network, facilitating the verification of its properties; (3) apply our method to more complex MAPF scenarios (e.g., with nonlinear agent dynamics and dynamically changing objectives) to thoroughly assess the effectiveness of our approach.

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (No. SWU-KQ23005), and National Natural Science Foundation of China (No. 62402400, No. 62272397, No. 62032019, No. 62302035).

References

Amir, O.; Sharon, G.; and Stern, R. 2015. Multi-Agent Pathfinding as a Combinatorial Auction. In Bonet, B.; and Koenig, S., eds., *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, 2003–2009. AAAI Press.

Athavale, A.; Bartocci, E.; Christakis, M.; Maffei, M.; Nickovic, D.; and Weissenbacher, G. 2024. Verifying Global Two-Safety Properties in Neural Networks with Confidence. In *CAV (2)*, volume 14682 of *Lecture Notes in Computer Science*, 329–351. Springer.

Black, M.; Fainekos, G.; Hoxha, B.; Prokhorov, D. V.; and Panagou, D. 2023. Safety Under Uncertainty: Tight Bounds with Risk-Aware Control Barrier Functions. In *IEEE International Conference on Robotics and Automation, ICRA 2023, London, UK, May 29 - June 2, 2023*, 12686–12692. IEEE.

Boffi, N. M.; Tu, S.; Matni, N.; Slotine, J. E.; and Sindhwani, V. 2020. Learning Stability Certificates from Data. In *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*, volume 155 of *Proceedings of Machine Learning Research*, 1341–1350. PMLR.

Borrmann, U.; Wang, L.; Ames, A. D.; and Egerstedt, M. 2015. Control Barrier Certificates for Safe Swarm Behavior. In *5th IFAC Conference on Analysis and Design of Hybrid Systems, ADHS 2015, Atlanta, GA, USA, October 14-16, 2015*, volume 48 of *IFAC-PapersOnLine*, 68–73. Elsevier.

Burian, S. J.; Velugubantla, S. P.; Chittineni, K.; Maddula, S. R. K.; and Brown, M. J. 2002. Morphological analyses using 3D building databases: Portland, Oregon. *Utah. LA-UR, Los Alamos National Laboratory, Los Alamos, NM, Tech. Rep*.

Cai, Z.; Cao, H.; Lu, W.; Zhang, L.; and Xiong, H. 2021. Safe Multi-Agent Reinforcement Learning through Decentralized Multiple Control Barrier Functions. *CoRR*, abs/2103.12553.

Chen, J.; Li, J.; Fan, C.; and Williams, B. C. 2021. Scalable and Safe Multi-Agent Motion Planning with Nonlinear Dynamics and Bounded Disturbances. In *AAAI*, 11237–11245. AAAI Press.

Chen, Y.; Peng, H.; and Grizzle, J. W. 2018. Obstacle Avoidance for Low-Speed Autonomous Vehicles With Barrier Function. *IEEE Trans. Control. Syst. Technol.*, 26(1): 194–206.

Cheng, R.; Khojasteh, M. J.; Ames, A. D.; and Burdick, J. W. 2020. Safe Multi-Agent Interaction through Robust Control Barrier Functions with Learned Uncertainties. In *59th IEEE Conference on Decision and Control, CDC 2020, Jeju Island, South Korea, December 14-18, 2020*, 777–783. IEEE.

Cheng, R.; Orosz, G.; Murray, R. M.; and Burdick, J. W. 2019. End-to-End Safe Reinforcement Learning through Barrier Functions for Safety-Critical Continuous Control Tasks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 3387–3395. AAAI Press.

Chu, W.; Zhang, W.; Zhao, H.; Jin, Z.; and Mei, H. 2023. Massive Shape Formation in Grid Environments. *IEEE Trans Autom. Sci. Eng.*, 20(3): 1745–1759.

Dimarogonas, D. V.; and Kyriakopoulos, K. J. 2007. On the Rendezvous Problem for Multiple Nonholonomic Agents. *IEEE Trans. Autom. Control.*, 52(5): 916–922.

Gu, S.; Kuba, J. G.; Chen, Y.; Du, Y.; Yang, L.; Knoll, A. C.; and Yang, Y. 2023. Safe multi-agent reinforcement learning for multi-robot control. *Artif. Intell.*, 319: 103905.

Han, L.; Sun, P.; Du, Y.; Xiong, J.; Wang, Q.; Sun, X.; Liu, H.; and Zhang, T. 2019. Grid-Wise Control for Multi-Agent Reinforcement Learning in Video Game AI. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, 2576–2585. PMLR.

Hinton, G. E.; Vinyals, O.; and Dean, J. 2015. Distilling the Knowledge in a Neural Network. *CoRR*, abs/1503.02531.

Hou, J.; Wang, F.; Wang, L.; and Chen, Z. 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*.

Jin, G.; Yi, X.; Zhang, L.; Zhang, L.; Schewe, S.; and Huang, X. 2020. How does Weight Correlation Affect Generalisation Ability of Deep Neural Networks? In *NeurIPS*.

Joe, W.; and Lau, H. C. 2023. Learning to Send Reinforcements: Coordinating Multi-Agent Dynamic Police Patrol Dispatching and Rescheduling via Reinforcement Learning. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, 153–161. ijcai.org.

Katz, G.; Huang, D. A.; Ibeling, D.; Julian, K.; Lazarus, C.; Lim, R.; Shah, P.; Thakoor, S.; Wu, H.; Zeljic, A.; Dill, D. L.; Kochenderfer, M. J.; and Barrett, C. W. 2019. The Marabou Framework for Verification and Analysis of Deep Neural Networks. In *CAV (1)*, volume 11561 of *Lecture Notes in Computer Science*, 443–452. Springer.

Khan, M.; Ibuki, T.; and Chatterjee, A. 2021. Safety Uncertainty in Control Barrier Functions using Gaussian Processes. In *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*, 6003–6009. IEEE.

Lin, W.; Yang, Z.; Chen, X.; Zhao, Q.; Li, X.; Liu, Z.; and He, J. 2019. Robustness Verification of Classification Deep Neural Networks via Linear Programming. In *CVPR*, 11418–11427. Computer Vision Foundation / IEEE.

Liu, J.; Zhan, N.; and Zhao, H. 2011. Computing semi-algebraic invariants for polynomial dynamical systems. In *EMSOFT*, 97–106. ACM.

Liu, Y.; and Zhang, X. 2022. Adaptive Random Testing for Multiagent Path Finding Systems. *IEEE Trans. Reliab.*, 71(1): 295–308.

Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; and Mordatch, I. 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.

Melcer, D.; Amato, C.; and Tripakis, S. 2022. Shield Decentralization for Safe Multi-Agent Reinforcement Learning. In *NeurIPS*.

Ondrácek, J.; Vanek, O.; and Pechoucek, M. 2015. Solving Infrastructure Monitoring Problems with Multiple Heterogeneous Unmanned Aerial Vehicles. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, 1597–1605. ACM.

Qin, Z.; Zhang, K.; Chen, Y.; Chen, J.; and Fan, C. 2021. Learning Safe Multi-agent Control with Decentralized Neural Barrier Certificates. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Barták, R.; and Boyarski, E. 2019. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. In *Proceedings of the Twelfth International Symposium on Combinatorial Search, SOCS 2019, Napa, California, 16-17 July 2019*, 151–159. AAAI Press.

Usevitch, J.; and Panagou, D. 2023. Adversarial Resilience for Sampled-Data Systems Under High-Relative-Degree Safety Constraints. *IEEE Trans. Autom. Control.*, 68(3): 1537–1552.

Wang, L.; Ames, A. D.; and Egerstedt, M. 2017. Safety Barrier Certificates for Collisions-Free Multirobot Systems. *IEEE Trans. Robotics*, 33(3): 661–674.

Zeng, X.; Yang, Z.; Zhang, L.; Tang, X.; Zeng, Z.; and Liu, Z. 2023. Safety Verification of Nonlinear Systems with Bayesian Neural Network Controllers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(12): 15278–15286.

Zhang, M.; Jin, Z.; Xu, Y.; Shen, Z.; Liu, K.; and Pan, K. 2021. Fast Adaptation to External Agents via Meta Imitation Counterfactual Regret Advantage. In *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, 1709–1711. ACM.

Zhang, X.-Y.; Liu, Y.; Arcaini, P.; Jiang, M.; and Zheng, Z. 2024. MET-MAPF: A Metamorphic Testing Approach for Multi-Agent Path Finding Algorithms. *ACM Trans. Softw. Eng. Methodol.*, 33(8).

Zhao, H.; Li, Q.; Zeng, X.; and Liu, Z. 2022. Safe Reinforcement Learning Algorithm and Its Application in Intelligent Control for CPS. *Int. J. Softw. Informatics*, 12(4): 453–483.

Zhao, H.; Zeng, X.; Chen, T.; Liu, Z.; and Woodcock, J. 2021. Learning safe neural network controllers with barrier certificates. *Formal Aspects Comput.*, 33(3): 437–455.

Zheng, Z.; and Gu, S. 2024. Safe Multi-Agent Reinforcement Learning with Bilevel Optimization in Autonomous Driving. *CoRR*, abs/2405.18209.

Zhou, Y.; Hu, H.; Liu, Y.; Lin, S.; and Ding, Z. 2019. A Real-Time and Fully Distributed Approach to Motion Planning for Multirobot Systems. *IEEE Trans. Syst. Man Cybern. Syst.*, 49(12): 2636–2650.

Zong, Z.; Zheng, M.; Li, Y.; and Jin, D. 2022. MAPDP: Cooperative Multi-Agent Reinforcement Learning to Solve Pickup and Delivery Problems. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, 9980–9988. AAAI Press.