LEARNING TO SPRAY IN UNCERTAIN AND WINDY ENVIRONMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deploying robotic systems for precision spraying offers a sustainable alternative to blanket spraying in agriculture. However, existing systems typically assume deterministic kinematics and perfect observations, ignoring the inherent stochasticity of agricultural fields such as unpredictable wind patterns, sensor noise, and variable spray-dispersion dynamics. To bridge this gap, we introduce a novel stochastic multi-robot environment for precision spraying that explicitly models real-world uncertainties: stochastic wind, actuation noise, observation noise, and probabilistic spray decay. We design a spraying mechanism based on wind-dependent Gaussian falloff that models advective spray drift and anisotropic plume elongation caused by variable wind patterns, and calibrate its parameters against published field measurements of UAV spray deposition. We formulate the task as a constrained stochastic optimization problem for coordinating robots to treat infected locations under uncertainty while minimizing path lengths and avoiding collisions. We constructed both centralized (single-agent) and decentralized (multi-agent) POMDPs that share the same global state space, dynamics, and reward function for performing this task. Our environment is compatible with standard single-agent and multi-agent deep Reinforcement Learning (RL) methods and serves as a rigorous benchmark for robust agricultural coordination, which can be modified or extended for other similar applications with real-world uncertainties. We evaluated centralized and decentralized deep RL methods and a model-based controller across multiple environment variations, demonstrating the feasibility of learning effective policies under various uncertainties.

1 INTRODUCTION

Effective pest and weed management is critical to agricultural productivity, yet traditional blanket spraying causes environmental degradation and health risks (Zhou et al., 2025; Mengistie et al., 2025; Shekhar et al., 2024; Varanasi et al., 2016). Selective spraying directly mitigates these issues; robotic spot-sprayers and UAVs have demonstrated chemical reductions of 35% to 60% while maintaining efficacy (Rahimi Azghadi et al., 2025; Allmendinger et al., 2022). The deployment of single and multi-robot systems for precision spraying therefore offers a highly promising path toward sustainable agriculture (Lal & Prabhakar, 2021; Lal et al., 2017; Baltazar et al., 2021; Ju et al., 2022; Rahimi Azghadi et al., 2025; Bzhikhatlov & Pshenokova, 2023). However, real-world deployment is fundamentally bottlenecked by *environmental uncertainties*. Wind disturbances alter robot trajectories and deflect spray droplets away from intended targets; actuator imprecision causes commanded thrusts to deviate from their nominal values; chemical deposition is inherently uncertain due to turbulence and canopy interaction; and onboard sensors provide only noisy estimates of robot states and infection levels. Ignoring these factors during environment design produces policies that are brittle when transferred to the field—a well-documented failure mode in applied reinforcement learning (RL) (Dulac-Arnold et al., 2021). Although much research has been done on the improvement of RL algorithms, the design and development of RL environments is equally important and presents unique challenges (Reda et al., 2020; Dulac-Arnold et al., 2021).

Many existing approaches to multi-robot precision spraying formulate the task as a combinatorial or mixed-integer program and compute open-loop plans under deterministic assumptions (Lal et al., 2017; Lal & Prabhakar, 2021; Xu et al., 2023; 2022; Li et al., 2022; Huang et al., 2024). While these methods are principled, they require the field geometry and infection locations to be fixed,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

scale poorly to high-dimensional continuous spaces, and provide no mechanism for adapting to disturbances encountered during execution. On the learning-based side, RL has recently been applied to single-robot (Hao et al., 2022; Khosravi et al., 2025) and multi-robot (Farid et al., 2025) precision spraying. However, broader surveys of RL in agricultural robotics (Goldenits et al., 2024; Ekechi et al., 2025) confirm that no existing work on multi-robot selective spraying jointly models wind, actuation noise, observation noise, and stochastic spray deposition, or supports both centralized and decentralized multi-agent execution. This paper fills that gap by formalizing efficient, customizable environments for multi-robot selective spraying with uncertainties. Our specific contributions are:

1. **Stochasticity and uncertainty.** We present cooperative multi-robot selective spraying environments that incorporate stochastic wind disturbance, spray-deposition noise, actuation noise, observation noise, and per-episode randomization.
2. **Multi-robot spraying mechanism.** We develop a wind-dependent multi-robot Gaussian spraying mechanism that models advective spray drift and anisotropic plume elongation caused by noisy wind patterns, discuss its relation to existing spray-drift models, and calibrate its parameters against published UAV field deposition data (Martin et al., 2025).
3. **Centralized, decentralized and model-based formulations.** We formulate the task as a constrained stochastic optimization problem, and represent it both as a centralized single-agent POMDP and as a decentralized multi-agent POMDP (Oliehoek & Amato, 2016). The two formulations share the same global state space, robot dynamics, and reward function, so any performance gap between them isolates the cost of decentralized execution rather than a change of task. We evaluate two single-agent (PPO (Schulman et al., 2017), TRPO (Schulman et al., 2015)) and two cooperative MARL baselines (MAPPO (Yu et al., 2022), VDN (Sunehag et al., 2018)), together with a model-based controller across different environment variants and team sizes. This directly compares centralized, decentralized, and model-based methods under identical stochastic and uncertain multi-robot spraying settings. The appendix adds a scalability study up to eight robots and extends the comparison to ten deep RL baselines.
4. **Physics-based 3-D simulation.** We simulate the environment on an agricultural field with herbicide-spraying UAVs using the CoppeliaSim robot simulator (Rohmer et al., 2013) with the Pybullet (Coumans & Bai, 2016–2021) physics engine, providing physics-based simulation beyond simple 2-D environments used in many existing RL works (Mnih et al., 2015; Haarnoja et al., 2018; Schulman et al., 2017; Towers et al., 2024).

2 Related Work

Selective spraying significantly reduces chemical waste compared to blanket application (Baltazar et al., 2021; Ju et al., 2022). Single-robot systems have evolved from GPS-guided maps (de Soto et al., 2016; Allmendinger et al., 2022) to real-time onboard deep learning detection, achieving up to 60% agrochemical reduction (Nasir et al., 2023; Rahimi Azghadi et al., 2025; Chang et al., 2024). However, these rely on fixed, rule-based controllers that lack dynamic adaptability. Multi-robot coordination typically utilizes optimization-based methods under discrete and/or deterministic assumptions (Lal & Prabhakar, 2021; Lal et al., 2017; Xu et al., 2023; 2022; Li et al., 2022; Huang et al., 2024). These open-loop approaches perform well in structured settings but lack the closed-loop adaptability needed to handle stochastic disturbances such as variable wind, actuation noise, or uncertain spray deposition.

Reinforcement learning (RL) has recently been applied to agricultural robotics for navigation, monitoring, and decision-making tasks (Goldenits et al., 2024). For spraying, Hao et al. (2022) proposed an adaptive spraying decision system that focuses on parameter optimization and does not consider multi-robot coordination or environmental stochasticity. Khosravi et al. (2025) uses a hierarchical PPO framework to jointly learn navigation and spraying decisions for a single robot without uncertainties. Farid et al. (2025) applied PPO to coordinate multiple aerial/ground vehicles for spraying with an explicit real-time wind-correction term. This formulation models only wind, and each vehicle is controlled by an independent single-agent policy rather than a jointly stochastic, partially observed, cooperatively coordinated task. Existing surveys of RL in agriculture and UAV systems (Goldenits et al., 2024; Ekechi et al., 2025) indicate that RL-based spraying applications are largely limited to simplified, discrete, or deterministic environments. In broader robotics, RL has achieved strong performance in multi-robot navigation and collision avoidance (Chen et al., 2017;

2021; Everett et al., 2018; Fan et al., 2020). These works focus on decentralized control and safe navigation, often in environments with static or dynamic obstacles, but do not address spraying tasks or consider such sources of environmental uncertainty. Recent advances in multi-agent reinforcement learning (MARL) provide scalable frameworks for decentralized coordination under uncertainty. Value decomposition methods such as VDN (Sunehag et al., 2018) and QMIX (Rashid et al., 2018) enable tractable learning by factorizing joint value functions, while actor–critic approaches such as MADDPG (Lowe et al., 2017) and MAPPO (Yu et al., 2022) achieve strong empirical performance in cooperative settings. These methods have been successfully applied to large-scale coordination problems. Hence, we evaluate them in an agricultural environment with stochastic disturbances. We believe that our proposed environment, along with its solution using centralized and decentralized RL methods, will give researchers a new direction to deploy RL-based systems that autonomously manage not just selective spraying, but a wide range of coordination tasks, leading to highly efficient and sustainable robotic systems.

3 PRELIMINARIES

Let $\mathbb{R}, \mathbb{R}_{\geq 0}, \mathbb{N}$ be the sets of real, non-negative real, and natural numbers, respectively. Let $\{x_i\}_{i=1}^n$ denote an ordered sequence or tuple of elements $x_1, x_2, \ldots, x_n$. For convenience, the L^2 or Euclidean norm $\|\cdot\|_2$ is simply denoted as $\|\cdot\|$. Let $\mathbf{I}_n$ denote the $n \times n$ identity matrix. Let $\mathrm{diag}(x_1, \ldots, x_n)$ denote the diagonal matrix with values $x_1, \ldots, x_n$. Let $\mathbf{0}_n$ and $\mathbf{1}_n$ denote the n-dimensional all-zeros and all-ones vectors. Let the clip function be $\mathrm{clip}(x, a, b) = \min(\max(x, a), b)$ for some $a \leq b$ (applied component-wise when x is a vector). Let $\mathbb{1}_C$ be a boolean variable where $\mathbb{1}_C = 1$ if condition C is true and 0 otherwise. For a set of samples S, let $\mathcal{P}(S)$ denote the set of probability distributions over S. For a distribution $\mathcal{X}$ with mean $\mu \in \mathbb{R}$ and variance $\sigma^2 \in \mathbb{R}_{\geq 0}$, a sample x drawn from this distribution is denoted as $x \sim \mathcal{X}(\mu, \sigma^2)$. Let $\mathcal{N}(\mu, \sigma^2)$ denote the univariate normal (Gaussian) distribution, and $\mathcal{N}_{[a,b]}(\mu, \sigma^2)$ its truncation to the interval $[a, b]$. Let $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denote the multivariate Gaussian distribution with mean vector $\boldsymbol{\mu} \in \mathbb{R}^n$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$, where $\boldsymbol{\Sigma}$ is symmetric positive semi-definite (Bishop, 2006). For a random variable Y, let $\mathbb{E}[Y]$ denote its expectation and $\mathrm{Var}(Y)$ its variance. Let $\Pr(E)$ denote the probability of an event E. In a fully observable Markov decision process (MDP), a policy $\pi : S \to \mathcal{P}(A)$ maps a state $s \in S$ to a probability distribution over the set of actions A. In a partially observable setting, let Ω be the observation space and let $\mathcal{H} = (\Omega \times A)^* \times \Omega$ denote the set of action–observation histories $h_t = (o_0, a_0, \ldots, a_{t-dt}, o_t)$. An *admissible* policy is a map $\pi : \mathcal{H} \to \mathcal{P}(A)$, equivalently a map from the belief $b_t(\cdot) = \Pr(s_t = \cdot \mid h_t) \in \mathcal{P}(S)$ to $\mathcal{P}(A)$, since the belief is a sufficient statistic of the history (Åström, 1965; Kaelbling et al., 1998). Following the standard notations and definitions (Le & Plaku, 2021; Choset et al., 2005), we define the multi-robot selective spraying problem in the next section.

4 PROBLEM FORMULATION

Consider a group of $n \in \mathbb{N}$ spraying robots $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$ operating in a bounded, d-dimensional workspace $\mathcal{W} \subseteq \mathbb{R}^d$. Each robot $\mathcal{M}_i$ is modeled as $\mathcal{M}_i = \{\mathcal{A}_i, \mathcal{Q}_i, \mathcal{S}_i, s_i^{\text{init}}, \mathcal{U}_i, f_i\}$ in terms of its shape $\mathcal{A}_i$, configuration space $\mathcal{Q}_i \subseteq \mathbb{R}^{d+1}$, state space $\mathcal{S}_i = \mathcal{Q}_i \times \mathcal{V}_i \subseteq \mathbb{R}^{2d+1}$ where $\mathcal{V}_i \subseteq \mathbb{R}^d$ is the velocity space, initial state $s_i^{\text{init}} \in \mathcal{S}_i$, control space $\mathcal{U}_i \subseteq \mathbb{R}^{d+1}$, and dynamics f_i (subject to stochastic disturbances). Each robot is an omnidirectional unmanned aerial vehicle (UAV) equipped with a spraying mechanism and a finite spray capacity $c_i \in \mathbb{R}_{\geq 0}$. Let $c_i^{\text{init}} > 0$ denote its initial spray capacity. The configuration $q_i = \langle p_i, c_i \rangle \in \mathcal{Q}_i$ denotes the position $p_i \in \mathbb{R}^d$ and spray capacity c_i of robot i. The state $s_i = \langle p_i, c_i, v_i \rangle \in \mathcal{S}_i$ denotes the operational state where $v_i \in \mathcal{V}_i$ is the velocity. The control input $u_i = \langle \alpha_i, \beta_i \rangle \in \mathcal{U}_i$ specifies the normalized thrust $\alpha_i \in [-1, 1]^d$ for changing the robot velocity, and the normalized spray signal $\beta_i \in [0, 1]$. The environment experiences a stochastic wind with magnitude $v_w(t) \in [0, v_w^{\max}]$ and direction $\theta_w(t)$, yielding a wind vector $\mathbf{w}(t) = \langle v_w(t) \cos\theta_w(t), v_w(t) \sin\theta_w(t) \rangle$ for $d = 2$ (and $\mathbf{w}(t) = \langle v_w(t) \cos\theta_w(t), v_w(t) \sin\theta_w(t), 0 \rangle$ for $d = 3$ by assuming a low-level altitude controller compensates gravity). A single wind vector $\mathbf{w}(t)$ is shared by the entire team. Real gusts persist over several seconds, so instead of resampling the wind independently at every timestep, we let both components follow a stationary first-order autoregressive process, the exact discretization of an

Ornstein–Uhlenbeck process (Uhlenbeck & Ornstein, 1930), with correlation time $\tau_w > 0$:

$$v_w(t+dt) = \bar{v}_w + \phi\big(v_w(t) - \bar{v}_w\big) + \sqrt{1-\phi^2}\,\sigma_w \xi_v(t), \tag{1}$$

$$\theta_w(t+dt) = \bar{\theta}_w + \phi\big(\theta_w(t) - \bar{\theta}_w\big) + \sqrt{1-\phi^2}\,\sigma_\theta \xi_\theta(t), \tag{2}$$

where $\phi = \exp(-dt/\tau_w)$, $\xi_v(t), \xi_\theta(t) \sim \mathcal{N}(0,1)$ are independent across time, $\bar{v}_w$ and $\bar{\theta}_w$ are the nominal wind magnitude and direction, and σ_w^2 and σ_θ^2 are the corresponding variances. The initial wind is drawn from the stationary marginals, $v_w(0) \sim \mathcal{N}_{[0,v_w^{\max}]}(\bar{v}_w, \sigma_w^2)$ and $\theta_w(0) \sim \mathcal{N}(\bar{\theta}_w, \sigma_\theta^2)$, and the magnitude is reflected at the bounds of $[0, v_w^{\max}]$, since a negative magnitude would silently reverse the wind vector. Setting $\phi = 0$ recovers independent per-step resampling, which we retain as an ablation. The actions and positions are perturbed by an action noise vector $\epsilon_a \sim \mathcal{N}(\mathbf{0}_d, \sigma_a^2 \mathbf{I}_d)$ and a position noise vector $\epsilon_p \sim \mathcal{N}(\mathbf{0}_d, \sigma_p^2 \mathbf{I}_d)$ where σ_a^2 and σ_p^2 are the action and position noise variances, respectively. Unlike the wind, these are drawn independently for each robot at each timestep. The relative airspeed $\mathbf{w}(t) - v_i(t)$ exerts a drag force scaled by an aerodynamic coefficient $k_w > 0$ and updates the velocity and position of robot i over a timestep dt as:

$$v_i(t+dt) = \mathrm{clip}\left(v_i(t) + \tfrac{k_a \alpha_i(t) + k_w(\mathbf{w}(t) - v_i(t))}{W_i} dt + \tfrac{\epsilon_a \sqrt{dt}}{W_i}, v_{\min}, v_{\max}\right), \tag{3}$$

$$p_i(t+dt) = p_i(t) + v_i(t)dt + \epsilon_p \sqrt{dt}, \tag{4}$$

where $W_i > 0$ is the mass of robot i, $k_a > 0$ is a thrust power coefficient, and $v_{\min} < v_{\max}$ are the bounds for preventing robots from gaining uncontrollable speed. The $\sqrt{dt}$ scaling of the noise terms makes Eqs. 3–4 an Euler–Maruyama discretization of the underlying diffusion (Kloeden & Platen, 1992). For selective spraying, the workspace $\mathcal{W}$ represents an agricultural field containing $m \in \mathbb{N}$ spatially distributed infection sites $\mathcal{I} = \{\mathcal{I}_1, \ldots, \mathcal{I}_m\}$. Each infected site $\mathcal{I}_j$ is associated with a fixed location $z_j \in \mathcal{W}$ and a normalized infection intensity $\ell_j(t) \in [0,1]$ that evolves over time. Let $\ell(t) = \{\ell_j(t)\}_{j=1}^m$ and $\ell^{\text{init}} = \{\ell_j(0)\}_{j=1}^m$ be the current and initial infection intensities of the m infected sites, respectively. To account for advective spray drift and anisotropic plume elongation caused by wind, let $\bar{t}_w = \mathbb{E}[t_{\text{fall}}]$, and $\sigma_{\text{dr}}^2 = \mathrm{Var}(t_{\text{fall}})$ be the expectation and variance of random spray droplet fall time $t_{\text{fall}} > 0$. We model the effective position of robot i as shifted by the wind, $\tilde{p}_i(t) = p_i(t) + \bar{t}_w \mathbf{w}(t)$. Furthermore, higher wind speeds elongate the spray pattern along the wind direction $\theta_w(t)$. For $d = 2$, we capture this by the dynamic covariance matrix,

$$\mathbf{\Sigma}_w(t) = \mathbf{R}(\theta_w(t)) \begin{bmatrix} \beta_s^2 + \sigma_{\text{dr}}^2 v_w(t)^2 & 0 \\ 0 & \beta_s^2 \end{bmatrix} \mathbf{R}(\theta_w(t))^\top, \tag{5}$$

where $\mathbf{R}(\theta_w(t)) = \begin{bmatrix} \cos\theta_w(t) & -\sin\theta_w(t) \\ \sin\theta_w(t) & \cos\theta_w(t) \end{bmatrix}$ is the 2-D rotation matrix (Choset et al., 2005) and $\beta_s > 0$ is a still-air spray dispersion parameter. Since $\mathbf{R}$ is orthogonal, $\mathbf{\Sigma}_w(t)$ is symmetric positive definite with the closed-form inverse $\mathbf{\Sigma}_w^{-1}(t) = \mathbf{R}(\theta_w(t))\,\mathrm{diag}\big((\beta_s^2 + \sigma_{\text{dr}}^2 v_w(t)^2)^{-1}, \beta_s^{-2}\big)\,\mathbf{R}(\theta_w(t))^\top$, so no numerical inversion is required at runtime. For $d = 3$ the wind is horizontal, and the elongation acts in the horizontal plane only: $\mathbf{\Sigma}_w(t) = \mathbf{R}_z(\theta_w(t))\,\mathrm{diag}\big(\beta_s^2 + \sigma_{\text{dr}}^2 v_w(t)^2,\ \beta_s^2,\ \beta_s^2\big)\,\mathbf{R}_z(\theta_w(t))^\top$ where $\mathbf{R}_z$ is the rotation about the vertical axis. Elongation redistributes a fixed volume of liquid over a larger footprint rather than creating additional deposit, so the peak deposition must fall as the plume spreads. We therefore define the wind-dependent gain $g_w(t) = \frac{\exp(-\eta_w v_w(t))\beta_s}{\sqrt{\beta_s^2 + \sigma_{\text{dr}}^2 v_w(t)^2}}$, where $\eta_w \geq 0$ is a wind attenuation coefficient. Spraying draws from a finite onboard tank, so the commanded signal is gated by the remaining capacity, $\beta_i^{\text{eff}}(t) = \min(\beta_i(t), c_i(t)/(k_\beta dt))$, where $k_\beta > 0$ is a spray coefficient. Thus, the effect of taking spray action β_i by robot i on infection site $\mathcal{I}_j$ at time t is modeled using a wind-dependent Gaussian falloff:

$$\hat{\Delta}\ell_j^i(t) = \mathrm{clip}\left(\beta_i^{\text{eff}}(t) \cdot \left(g_w(t) \exp\left(-\tfrac{1}{2}(z_j - \tilde{p}_i(t))^\top \mathbf{\Sigma}_w^{-1}(t)(z_j - \tilde{p}_i(t))\right) dt + \epsilon_s \sqrt{dt}\right), 0, \ell_j(t)\right), \tag{6}$$

where $\epsilon_s \sim \mathcal{N}(0, \sigma_s^2)$ is the spray noise with variance σ_s^2 (this noise varies for every robot at every timestep, but is written simply as ϵ_s for convenience). We rescale the nominal depositions of robot i as $\Delta\ell_j^i(t) = \chi_i(t)\,\hat{\Delta}\ell_j^i(t)$ with $\chi_i(t) = \min\big(1, k_\beta \beta_i^{\text{eff}}(t) dt / \sum_j \hat{\Delta}\ell_j^i(t)\big)$ if $\sum_j \hat{\Delta}\ell_j^i(t) \neq 0$ and 1

otherwise, so that the total deposition of robot i never exceeds $k_\beta \beta_i^{\text{eff}}(t)dt$. Each infection level ℓ_j and capacity c_i then evolve as:

$$\ell_j(t+dt) = \max(0, \ell_j(t) - \sum\nolimits_{i=1}^{n} \Delta\ell_j^i(t)), \tag{7}$$

$$c_i(t+dt) = \max(0, c_i(t) - k_\beta \beta_i^{\text{eff}}(t)dt). \tag{8}$$

Note that the reduction of capacity $c_i(t)$ does not depend on wind speed or infection reduction; it only depends on the spray coefficient k_β and the effective spray signal $\beta_i^{\text{eff}}(t)$ at time t. In still air ($v_w = 0$), we have $g_w = 1$ and $\mathbf{\Sigma}_w = \beta_s^2 \mathbf{I}_d$, and Eq. 6 reduces to the isotropic kernel $\exp(-\|z_j - p_i(t)\|^2 / 2\beta_s^2)$.

Relation to established drift models: Eqs. 5–6 are deliberately a lightweight surrogate for the mechanistic spray-drift models used in the agricultural engineering and regulatory literature. AGDISP (Bilanin et al., 1989) and its regulatory derivative AgDRIFT (Teske et al., 2002) track droplet populations with a Lagrangian formulation in the near field and transition to a Gaussian plume description further downwind, while FSCBG (Dumbauld et al., 1980) and the simplified predictors of Craig et al. (1998) work directly with Gaussian dispersion. Our kernel keeps precisely the structure those models share: a Gaussian deposition footprint whose center advects with the wind and whose covariance grows along the wind direction, yet it remains cheap enough to evaluate at every RL timestep for every robot–site pair, which a Lagrangian particle simulation is not. This also determines how the parameters should be set, rather than being chosen by hand: $\beta_s, \bar{t}_w, \sigma_{\text{dr}}$ and η_w can be fitted by least squares to deposition curves generated by AGDISP/AgDRIFT for a given nozzle and release height, or to field measurements collected under the ISO 22866 protocol (ISO, 2005). AGDISP and AgDRIFT could not be obtained for this study, so we fit the kernel to the published field deposition model of Martin et al. (2025), measured downwind of an 8-rotor UAV spraying at a 3 m release height under 0.4–3.2 m/s wind, evaluated at downwind distances of 3.5–20.5 m for 0.5–3.0 m/s, the nominal wind band of our environments. The fit gives $\beta_s = 3.59$ m and $\bar{t}_w = 0.80$ s ($R^2 = 0.94$ on the deposited-mass scale), while σ_{dr} and η_w are not identified by these data and are set to zero. The calibrated kernel is therefore an isotropic Gaussian footprint advected by $\bar{t}_w \mathbf{w}(t)$; the elongation and attenuation terms remain available for references that identify them (Appendix C.3).

The updates in Eq. 3–4 and 7–8 follow the same discretization, which reduces to the standard forward Euler scheme (Åström & Murray, 2020) when all noise variances vanish. From a state $s_i(t) = \langle p_i, c_i, v_i \rangle$ at time t, robot i can reach a new state by applying a control $u_i(t) = \langle \alpha_i, \beta_i \rangle$ for one step dt:

$$s_i(t+dt) \leftarrow \text{SIM}(s_i(t), u_i(t), f_i, dt; \mathbf{w}(t), \epsilon(t)), \tag{9}$$

which numerically integrates f_i by following the dynamics defined in Eq. 3, 4 and 8. The exogenous wind $\mathbf{w}(t)$ and the noise draws $\epsilon(t) = \langle \epsilon_a, \epsilon_p, \epsilon_s \rangle$ are added explicitly to emphasize that SIM is a stochastic map. Starting at a state $s_i^{\text{init}} \in \mathcal{S}_i$ and applying a sequence of controls in succession, each for a time step dt, gives rise to a dynamically-feasible trajectory $\zeta_i : [0, T] \to \mathcal{S}_i$ where at any $t \in [0, T]$ we have:

$$\zeta_i(t+dt) \leftarrow \text{SIM}(\zeta_i(t), u_i(t), f_i, dt; \mathbf{w}(t), \epsilon(t)). \tag{10}$$

Let us define a cost function $\mathcal{J} : \mathcal{S}_i \times \mathcal{U}_i \times [0,1]^m \to \mathbb{R}_{\geq 0}$ where for a state $s_i(t) \in \mathcal{S}_i$, control $u_i(t) \in \mathcal{U}_i$, and infection levels $\ell(t)$ at time t:

$$\mathcal{J}(s_i(t), u_i(t), \ell(t)) = \lambda_1 \|\alpha_i(t)\|^2 + \lambda_2 \|v_i(t)\| + \frac{\lambda_3}{n} \sum_{j=1}^{m} \ell_j(t) + \frac{\lambda_4}{n} \tag{11}$$

where $\lambda_1 \|\alpha_i(t)\|^2$ is the energy cost, $\lambda_2 \|v_i(t)\|$ is the movement cost, $\frac{\lambda_3}{n} \sum_{j=1}^{m} \ell_j(t)$ is the infection cost, $\frac{\lambda_4}{n}$ is the step cost, and $\lambda_1, \lambda_2, \lambda_3, \lambda_4 \in \mathbb{R}_{\geq 0}$ are some fixed weights. The objective is to jointly control all robots to eliminate all infection sites while minimizing the costs and avoiding inter-robot collisions. Because the dynamics are stochastic, the decision variable is a feedback policy rather than a fixed control sequence. Moreover, because some noises are unbounded, no policy can guarantee containment or collision avoidance with probability one. We therefore state the constraints as chance constraints with a small tolerance $\delta \in (0, 1)$. Let Π be the set of *admissible* policies, and $u(t) = \langle u_1(t), \ldots, u_n(t) \rangle \sim \pi(\cdot \mid h_t)$. Therefore, we formulate the following stochastic constrained

optimization problem:

$$\min_{\pi\in\Pi} \mathbb{E}_\pi\Big[\sum_{t=0}^{T}\sum_{i=1}^{n} \mathcal{J}(\zeta_i(t), u_i(t), \ell(t)) \mid \zeta_i(0) = s_i^{\text{init}}, \ell(0) = \ell^{\text{init}}\Big] \tag{12}$$

s.t.,

$$\text{(i)}\ \Pr_\pi[\textstyle\sum_{j=1}^{m} \ell_j(T) < \ell_\epsilon] > 1-\delta, \tag{13}$$

$$\text{(ii)}\ \Pr_\pi[\forall t \in [0,T], \forall \mathcal{M}_i \in \mathcal{M} : p_i(t) \in \mathcal{W}] > 1-\delta, \tag{14}$$

$$\text{(iii)}\ \Pr_\pi[\forall t \in [0,T], \forall (\mathcal{M}_i \neq \mathcal{M}_k) : \mathcal{A}_i(t) \cap \mathcal{A}_k(t) = \emptyset] > 1-\delta, \tag{15}$$

where $\ell_\epsilon \approx 0$ is an infection-completion threshold, and $\mathcal{A}_i(t), \mathcal{A}_k(t) \subset \mathcal{W}$ are the regions occupied by robots i, k at time t in $\mathcal{W}$. Here, the first constraint ensures that all m infected locations are sprayed. The second constraint ensures that the robots remain within the workspace. And the third constraint ensures that there are no collisions. The definition of ζ ensures that the robots follow the dynamics. We assume the instance is feasible, i.e., that there exists some policy $\pi' \in \Pi$ which satisfies all three constraints given $\{s_i^{\text{init}}\}_{i=1}^n$, ℓ^{init} and $\mathbf{w}(0)$.

5 Environment Design

The optimization problem is expressed in two ways: as a centralized single-agent POMDP, which is used by the standard deep RL baselines, and as a decentralized multi-agent POMDP (Dec-POMDP), which is used by MARL baselines.

5.1 Centralized single-agent POMDP

The centralized single-agent POMDP is defined as $M = (S, A, \Omega, O, \mathcal{T}, R, b_0, \gamma)$. The **State Space** $S = \mathcal{S}_1 \times \ldots \times \mathcal{S}_n \times [0,1]^m \times \mathbb{R}^d \subseteq \mathbb{R}^{(2d+1)n+m+d}$ captures the joint state $s_t = (\{s_i(t)\}_{i=1}^n,\ \{\ell_j(t)\}_{j=1}^m, \mathbf{w}(t))$, and $b_0 \in \mathcal{P}(S)$ is the initial state distribution induced by the sampling of $\{s_i^{\text{init}}\}_{i=1}^n$, ℓ^{init} and $\mathbf{w}(0)$. The **Action Space** $A = \mathcal{U}_1 \times \ldots \times \mathcal{U}_n$ captures joint actions $a_t = \langle u_1, \ldots, u_n\rangle$ where $u_i = \langle \alpha_i, \beta_i\rangle$. To model sensor inaccuracies, the **Observation Space** Ω provides a noisy estimate $o_t \in \Omega$ of the partially normalized state together with an onboard wind measurement:

$$o_t = \Big(\{(p_i(t), \tilde{c}_i(t), \tilde{v}_i(t))\}_{i=1}^n,\ \{\ell_j(t)\}_{j=1}^m,\ \mathbf{w}(t)\Big) + \epsilon_o(t), \tag{16}$$
$$\epsilon_o(t) \sim \mathcal{N}\big(\mathbf{0}_{d_o},\ \text{diag}(\sigma_{obs}^2 \mathbf{I}_{(2d+1)n+m},\ \sigma_{w,obs}^2 \mathbf{I}_d)\big),$$

where $d_o = (2d+1)n + m + d$, $\tilde{c}_i(t) = c_i(t)/c_i^{\text{init}}$, $\tilde{v}_i(t) = (v_i(t) - v_{\min}\mathbf{1}_d)/(v_{\max} - v_{\min})$, σ_{obs}^2 is the joint variance of position, capacity, velocity and infection-levels, and the last d components of $\epsilon_o(t)$ model onboard anemometer noise for measuring $\mathbf{w}(t)$. The joint observation noise $\epsilon_o(t)$ is drawn independently at each timestep. The **Observation Function** $O : S \to \mathcal{P}(\Omega)$ maps each true state to a distribution over observations, realized by adding Gaussian noise as in Eq. 16. The **Transition Function** $\mathcal{T} : S \times A \to \mathcal{P}(S)$ governs the stochastic dynamics, including wind and action noise via Eq. 3–4, and infection/capacity updates via Eq. 7–8. The wind component of the state is exogenous: $\mathbf{w}(t+dt)$ drawn from Eqs. 1–2 given $\mathbf{w}(t)$, independently of the remaining components of s_t and a_t. Because $\mathbf{w}(t)$ is part of s_t, the temporally correlated wind keeps the process Markov, and the anemometer entry of o_t reports the wind that acts during the next transition. The **Reward Function** $R : S \times A \times S \to \mathbb{R}$ balances infection reduction against collision avoidance and path efficiency (the three-argument reward function (Sutton & Barto, 2018) is used because the next state is noisy). Let $t' = t + dt$ and $r_s, r_L \in \mathbb{R}_{\geq 0}$ be some small and very large reward values where

$r_s \ll r_L$. The total reward is:

$$R(s_t, a_t, s_{t'}) = r_{\text{spray}} + r_{\text{term}} + r_{\text{path}} \tag{17}$$

$$r_{\text{spray}} = r_s \sum_{j=1}^{m} (\ell_j(t) - \ell_j(t')) \tag{18}$$

$$r_{\text{term}} = \begin{cases} -r_L, & \text{if } \exists(\mathcal{M}_i \neq \mathcal{M}_k) : \mathcal{A}_i(t') \cap \mathcal{A}_k(t') \neq \emptyset \\ +r_L, & \text{else if } \sum_{j=1}^{m} \ell_j(t') < \ell_\epsilon \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

$$r_{\text{path}} = -\sum_{i=1}^{n} \mathcal{J}(s_i(t), u_i(t), \ell(t)) - r_s \lambda_b \sum_{i=1}^{n} \mathbb{1}_{p_i(t') \notin \mathcal{W}} \tag{20}$$

Here, $\lambda_b > 0$ is a scaling parameter used for penalizing the robots going outside the field, and r_L is chosen to be bigger than any cumulative summation of r_s (with or without scaling) over the time horizon. Finally, the **Discount Factor** $\gamma \in [0, 1)$ incentivizes rapid infection elimination. Setting $\sigma_{obs} = 0$ and $\sigma_{w,obs} = 0$ makes O the identity and reduces M to the fully observed MDP $\bar{M} = (S, A, \mathcal{T}, R, b_0, \gamma)$, whose optimal policy is a state-feedback law $\pi^* : S \to \mathcal{P}(A)$. For positive observation noise, an admissible policy in the sense of Section 3 acts on the belief b_t, equivalently on the history h_t. The learned policies benchmarked in Section 6 are memoryless and act on o_t alone; they therefore search a strict subset of Π, and their returns should be read as evidence of learnability under uncertainty.

5.2 Decentralized multi-agent POMDP

To model decentralized execution where robots rely solely on local information, we extend the centralized POMDP into a Decentralized Partially Observable Markov Decision Process (Dec-POMDP). The Dec-POMDP is defined by the tuple $M_{dec} = (\mathcal{M}, S, \{A_i\}_{\mathcal{M}_i \in \mathcal{M}}, \{\Omega_i\}_{\mathcal{M}_i \in \mathcal{M}}, \{O_i\}_{\mathcal{M}_i \in \mathcal{M}}, \mathcal{T}, R, b_0, \gamma)$. The joint state space S, transition dynamics $\mathcal{T}$, global reward function R, initial state distribution b_0, and discount factor γ remain identical to the centralized formulation, as the underlying physics of the environment and the cooperative objective of the team do not change. The distinction lies in the factorization of actions and observations. The joint action space factors into local action spaces $A = \prod_{i=1}^{n} A_i$, where $A_i = \mathcal{U}_i$. At each timestep, robot i measures its own internal state, estimates the infection intensities, senses the wind through its own anemometer, and detects other robots within an onboard sensing radius $\rho_{\mathcal{M}} > 0$. For two robots $\mathcal{M}_i \neq \mathcal{M}_k$, let $\mu_{i,k}(t) = \mathbb{1}_{\|p_k(t) - p_i(t)\| \leq \rho_{\mathcal{M}}}$ be the neighbor visibility mask, and let $\phi_{i,k}(t) = \big(p_k(t) - p_i(t),\ \tilde{v}_k(t) - \tilde{v}_i(t),\ \mu_{i,k}(t)\big)$ be the relative feature block for neighbor k. Let $\mathcal{K}_i(t)$ collect the ν nearest robots with $\mu_{i,k}(t) = 1$, ordered by distance and zero-padded to exactly ν slots. The local observation is

$$o_{i,t} = \Big(p_i(t),\ \tilde{c}_i(t),\ \tilde{v}_i(t),\ \{\ell_j(t)\}_{j=1}^{m},\ \mathbf{w}(t),\ \{\phi_{i,k}(t)\}_{k \in \mathcal{K}_i(t)}\Big) + \epsilon_{o,i}(t), \tag{21}$$

where $\epsilon_{o,i}(t) \sim \mathcal{N}(\mathbf{0}_{d_i}, \mathbf{\Sigma}_{obs,i})$ with $\mathbf{\Sigma}_{obs,i}$ diagonal and zero on the binary mask entries, and $d_i = (2d+1) + m + d + \nu(2d+1)$. Because the neighbor budget ν is fixed, $\dim(\Omega_i)$ does not grow with the team size n. The neighbor block is what makes the collision term of Eq. 19 learnable under decentralized execution. Each **Observation Function** $O_i : S \to \mathcal{P}(\Omega_i)$ is the distribution induced by Eq. 21, and the joint observation function factorizes as $O(\vec{o} \mid s) = \prod_{i=1}^{n} O_i(o_i \mid s)$, assuming conditionally independent onboard sensors, where $\vec{o} = \langle o_1, \ldots, o_n \rangle$. Consequently, execution is fully decentralized. Each robot makes decisions based only on its local action–observation history $h_{i,t} = (o_{i,0}, a_{i,0}, \ldots, a_{i,t-dt}, o_{i,t}) \in \mathcal{H}_i$. The goal of the Dec-POMDP is to find a joint policy $\vec{\pi} = \langle \pi_1, \ldots, \pi_n \rangle$, where each local policy $\pi_i : \mathcal{H}_i \to \mathcal{P}(A_i)$ maps the local history to a probability distribution over local actions, such that the expected global cumulative reward is jointly maximized.

6 Experiments

We evaluate centralized and decentralized RL methods along with a model-based controller across different team sizes ($N \in \{2, 3, 4, 5\}$) and 10 environment variations. We compare single-agent baselines on the centralized POMDP (PPO (Schulman et al., 2017), TRPO (Schulman et al., 2015)) against cooperative MARL baselines on the Dec-POMDP (MAPPO (Yu et al., 2022), VDN (Sunehag

Table 1: IQM of episodic return (50 fresh episodes per run) over 5 seeds × 10 environment variations; Table 13 adds the 95% stratified-bootstrap confidence intervals. Hyperparameters are tuned at $N = 3$ only and transferred to $N \in \{2, 4, 5\}$ without re-tuning.

Algorithm	Default hyperparameters				Tuned hyperparameters			
	$N = 2$	$N = 3$	$N = 4$	$N = 5$	$N = 2$	$N = 3$	$N = 4$	$N = 5$
PPO	**3491**†	**3514**†	3287†	−255	**3484**†	**3534**†	3549†	**3527**†
TRPO	−21059	−25265	−16220	−1438	3365†	1755†	−858	−1516
MAPPO	2127†	3510†	**3543**†	**3549**†	2310†	3533†	**3551**†	3519†
VDN	2927†	3451†	3455†	3416†	2369†	3144†	3335†	2626†
Model-based	2042†	2789†	2479†	2072†	2651†	2926†	2894†	1984†

Table 2: Ablation results at $N = 3$ with a 90% completion threshold for tuned MAPPO, VDN, and the model-based controller. Reward ablation: IQM of episode length and % of episodes ending by sprayed / collision. Observation and uncertainty ablation reports IQM of return.

Reward ablation

Condition	MAPPO		VDN		Model-based	
	Ep. len. IQM	Sprayed / coll. (%)	Ep. len. IQM	Sprayed / coll. (%)	Ep. len. IQM	Sprayed / coll. (%)
Full reward	**17.6**	**100 / 0**	**22.7**	95 / 4	**54.3**	96 / 4
No r_{term}	18.1	83 / 17	28.8	**96 / 4**	54.6	96 / 4
No r_{spray}	17.2	93 / 5	41.1	88 / 10	54.6	96 / 4
No r_{path}	63.3	83 / 1	395.9	56 / 11	54.6	96 / 4

Observation ablation

Condition	Return IQM		
	MAPPO	VDN	Model-based
Full	**3416**†	**3353**†	2921†
No pos.	−28935	−1778	2951†
No infect.	3410†	3349†	**3171**†
Pos. only	3388†	234	3163†

Uncertainty ablation

Condition	Return IQM		
	MAPPO	VDN	Model-based
Full	3416†	3353†	2921†
Wind only	3426†	3346†	3073†
Act. only	3418†	3303†	2990†
Obs. only	**3429**†	**3370**†	2960†
Deterministic	3423†	3338†	**3102**†

et al., 2018)). An extended 10-learner benchmark is provided in Appendix E.3. A classical model-based controller serves as a non-learning reference (Appendix D). Each RL model trains for 2×10^6 steps across 5 seeds. The best checkpoint of each run is re-evaluated on 50 held-out episodes whose seeds never occur in training or tuning and are shared by all methods. Scores reflect the interquartile mean (IQM) (Agarwal et al., 2021) of the mean returns. We evaluate both default and Optuna-tuned hyperparameters. Tuning occurs only at $N = 3$ (using 3 seeds × 3 distinct variations) with a TPE sampler (Bergstra et al., 2011) and median pruning, and transfers to other team sizes without re-tuning. For all experiments, any return IQM > 1700 certifies full spray coverage. Hence, for all tables, **Bold** indicates the best result per column or block, and † marks the cells above the success threshold. For the model-based controller, the default and tuned columns use its untuned and tuned parameters, respectively. More details and the full experimental protocol are given in Appendix E.

6.1 Main Results

Table 1 reports the held-out IQM of episodic returns. For default hyperparameters, decentralized learners succeed at every team size, with MAPPO best or tied at $N \geq 3$ (3510–3549). Centralized learners degrade as the team grows due to expanding joint observation/action spaces. PPO succeeds up to $N = 4$ but collapses at $N = 5$ (-255; 63% collision rate). TRPO never succeeds, experiencing 62–76% collision rates. For decentralized learners, $N = 2$ is the hardest configuration as two robots require the most time to cover the field, often hitting the step limit. For tuned hyperparameters, tuning at $N = 3$ improves centralized learners. Tuned PPO succeeds at all sizes, reaching 3484–3549 at every N. Tuned TRPO clears the threshold at $N \in \{2, 3\}$ but still fails at $N \geq 4$. Tuned MAPPO stays within 30 of its default or above it at every N, completing $\geq 99\%$ of episodes at $N \geq 3$ with negligible collisions. Conversely, tuned VDN scores *lower* than its default (e.g., 3144 vs 3451 at $N = 3$), indicating the tuning configuration overfit the three tuning variations. The classical controller reliably completes the task at every N (1984–2926), but scores 600–1550 points below the best RL methods.

6.2 Ablation Study

We ablate reward terms, observation blocks, and uncertainty sources at $N = 3$ for MAPPO and VDN with their tuned hyperparameters and for the tuned classical controller (Table 2), with the spray completion threshold set to 90%. The reward and observation ablations use 3 seeds × 10 variations per condition, and the uncertainty ablation uses 5 seeds × 10 variations. The Full rows of all three blocks are one shared set of runs (5 seeds × 10 variations), trained once under the unablated configuration. All cells are scored on the held-out episodes described above. For reward ablations, each term is critical. Removing r_{term} spikes MAPPO's collision rate from 0% to 17% and slows VDN. Removing r_{spray} drops completion rates for both learners. Removing r_{path} eliminates the

Table 3: Observation-gap analysis at $N = 3$: IQM of return under inference-time sensor perturbations for the tuned MAPPO, VDN and classical controller. Δ: mean per-run relative change against each run's unperturbed baseline. **Bold**: best perturbed condition per column.

Perturbation	MAPPO		VDN		Model-based	
	IQM	Δ(%)	IQM	Δ(%)	IQM	Δ(%)
None (baseline)	$3532^{\dagger}$	—	$3084^{\dagger}$	—	$3009^{\dagger}$	—
Position noise ($\sigma = 5.0$ m)	$3503^{\dagger}$	-0.9	$2494^{\dagger}$	-146.7	10	-98.9
Velocity staleness (3 steps)	$3528^{\dagger}$	-0.1	$2631^{\dagger}$	-176.2	$1787^{\dagger}$	-46.7
Field-map loss (10%)	$\mathbf{3532}^{\dagger}$	$\mathbf{+0.0}$	$\mathbf{2990}^{\dagger}$	$\mathbf{-43.3}$	$\mathbf{3020}^{\dagger}$	$\mathbf{-0.1}$
All combined (staleness 5)	$3475^{\dagger}$	-1.9	1316	-329.8	-1261	-143.5



(a) Var. 1 init (b) Var. 1 final (c) Var. 3 init (d) Var. 3 final

Figure 1: 3-D simulation in PyBullet for $N = 3$ in two variations (Var. 1 and 3). Dark green patches denote infection sites; red lines denote spraying paths.

incentive for efficiency, drastically inflating episode lengths and causing VDN to complete only 56% of its episodes. For observation ablations, self-position is indispensable. Without it, MAPPO and VDN returns collapse. Surprisingly, omitting the infection map barely impacts the learners (MAPPO 3410, VDN 3349), as they memorize the layout of the variations rather than reading the map. MAPPO succeeds using *only* positions (3388), whereas VDN fails (234). The classical controller actually improves without the noisy observed map (2921 $\rightarrow$ 3171), as its internal model-based prediction serves it better. For uncertainty ablations, calibrated noise barely affects the learners. All uncertainty conditions yield stable, overlapping returns (MAPPO 3416–3429, VDN 3303–3370).

6.3 Simulation and Observation Gap

We deployed the policies in PyBullet (Fig. 1) and injected inference-time sensor perturbations into the observation (Table 3). The perturbations target the same components for every method: each robot's own position (additive Gaussian noise with $\sigma = 5.0$ m, against the calibrated still-air dispersion $\beta_s = 3.59$ m and the 2.0 m collision distance), its own velocity (reported with a 3-step lag), and the shared infection map (lost entirely with probability 0.1 at each step), and finally all three together with a 5-step lag. Under decentralized execution each robot's sensors are corrupted independently. The study covers all 50 tuned $N = 3$ runs of MAPPO and VDN and the tuned classical controller on the same (variation, seed) cells, with 100 held-out episodes per run and condition; episode seeds and corruption streams are shared across conditions and methods. MAPPO is almost unaffected: its largest drop is 1.9%, under all perturbations combined. VDN degrades far more. Its mean relative changes in Table 3 are inflated by a few runs whose baseline return is near zero or negative, and by a few runs that collapse; relative to the baseline IQM, its drops are 19% (position noise), 15% (staleness), 3% (map loss) and 57% (combined). The classical controller is the most fragile: it steers by the observed position, so 5 m position noise leaves it at an IQM of 10 (-98.9%), and the combined condition drives it to -1261. The map loss hurts no method much, which agrees with the observation ablation: the policies rely little on the observed map. This experiment proves the superiority of RL over model-based controllers under different perturbations.

7 Conclusion

We introduced multi-robot environments for selective agricultural spraying, featuring a wind-dependent deposition kernel that models advective spray drift and anisotropic plume elongation. Our experimental results demonstrate that decentralized execution is highly competitive with the centralized one, which is important since in real deployment, the robots may only have access to the local states instead of the global states. Our future plan includes deploying and testing this model in physical UAVs.

AI USE STATEMENT

In this work, we used generative AI tools for extending experiments, assessing the literature, and critiquing. The original multi-robot environments and simple training scripts were written manually by the authors in single-file implementations, and AI tools were used to extend them for performing full experiments and ablation studies. We used LLMs to collect the literature survey, but manually checked and corrected all citations. We used AI tools to generate reviews and critiques of our paper, and manually edited the paper to incorporate the suggestions that were needed. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work introduces a simulated environment and a set of reinforcement learning baselines; it involves no human subjects, no personally identifiable information, and no newly collected data. The ten environment variations are synthetic and are released with the code.

The application domain does carry dual-use considerations that we think are worth stating plainly. The task we optimize is the *reduction* of agrochemical use: a policy that sprays selectively applies herbicide only near infected sites, which is the mechanism by which robotic spot-spraying achieves the 35–60% chemical reductions reported in the literature we cite in Section 1. The same control policy, pointed at a different objective, could distribute any liquid payload. We consider the risk comparable to that of the agricultural robotics platforms our environment models, and lower than that of the field-deployed systems those platforms already represent, since nothing here is tied to physical hardware.

We would, however, caution against reading our results as evidence of field-readiness. The observation-gap study of Section 6 measures how policies that perform well in simulation degrade under sensor perturbation (the decentralized MAPPO policies lose at most 2% of their return, whereas VDN loses up to 57% of its baseline IQM and the classical controller fails outright under 5 m position noise). Deploying a policy of this kind on a vehicle carrying agricultural chemicals would require safety machinery that this paper does not provide: the collision and containment constraints of Eqs. 14–15 are enforced only in expectation, through the reward, and are not certified. Section A states this limitation in full.

REPRODUCIBILITY STATEMENT

The environment is the contribution of this paper, so we have tried to make it reproducible from the text alone and then, independently, from the code.

Every quantity that Sections 4 and 5 leave symbolic is given a value in Table 4, including the calibrated parameters of the deposition kernel of Eqs. 5–6 (fitted as described in Section C.3), the wind process of Eqs. 1–2, and the reward weights of Eqs. 17–20. The ten environment variations are listed in Table 6, the observation layouts of Eqs. 16 and 21 in Table 7, the discretization used by the value-factorization baseline in Section C.7, and the classical controller in Section D.

The experimental protocol — seeds, step budget, hardware, checkpoint selection, held-out evaluation and the statistical treatment of the results — is given in Section E.1. Hyperparameters are reported for both regimes and both formulations: the default values in Section E.1 and Table 12, the search spaces in Tables 9 and 8, and the selected configurations in Tables 10 and 11.

The source code is submitted as supplementary material and is described in Section E.7. It contains the environment in both formulations, the multi-agent baselines, the classical controller, the spray-kernel calibration, one entry point per stage of the pipeline, a single experiment plan from which both a one-machine scheduler and the cluster job lists are generated, and the analysis script that produces every table from the per-run logs. The decentralized environment is built by composition from the centralized one, so the two formulations share their state space, transition function, reward and discount factor by construction rather than by assertion.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Alicia Allmendinger, Michael Spaeth, Marcus Saile, Gerassimos G. Peteinatos, and Roland Gerhards. Precision chemical weed management strategies: A review and a design of a new cnn-based modular spot sprayer. *Agronomy*, 12(7), 2022. ISSN 2073-4395. doi: 10.3390/agronomy12071620. URL https://www.mdpi.com/2073-4395/12/7/1620.

Karl Johan Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965. doi: 10.1016/0022-247X(65)90154-X.

André Rodrigues Baltazar, Filipe Neves dos Santos, António Paulo Moreira, António Valente, and José Boaventura Cunha. Smarter robotic sprayer system for precision agriculture. *Electronics*, 10 (17):2061, 2021.

J. Benton, Amanda Coles, and Andrew Coles. Temporal planning with preferences and time-dependent continuous costs. *Proceedings of the International Conference on Automated Planning and Scheduling*, 22:2–10, May 2012. ISSN 2334-0835. doi: 10.1609/icaps.v22i1.13509. URL http://dx.doi.org/10.1609/icaps.v22i1.13509.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

A Bhatt, D Palenicek, B Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and J Peters. Crossq: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *International Conference on Learning Representations (ICLR)*, 2024.

Alan J. Bilanin, Milton E. Teske, John W. Barry, and Robert B. Ekblad. AGDISP: The aircraft spray dispersion model, code development and experimental validation. *Transactions of the ASAE*, 32 (1):327–334, 1989. doi: 10.13031/2013.31005.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, NY, 2006. ISBN 978-0-387-31073-2.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016. URL https://arxiv.org/abs/1606.01540.

Hoang-Dung Bui, Yuanjie Lu, and Erion Plaku. Improving the efficiency of sampling-based motion planners via runtime predictions for motion-planning problems with dynamics. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4486–4491, 2022. doi: 10.1109/IROS47612.2022.9981753.

Kantemir Bzhikhatlov and Inna Pshenokova. Intelligent spraying system of autonomous mobile agricultural robot. In Andrey Ronzhin and Alexander Kostyaev (eds.), *Agriculture Digitalization and Organic Production*, pp. 269–278, Singapore, 2023. Springer Nature Singapore. ISBN 978-981-99-4165-0.

Chung-Liang Chang, Hung-Wen Chen, and Jing-Yun Ke. Robust guidance and selective spraying based on deep learning for an advanced four-wheeled farming robot. *Agriculture*, 14(1), 2024. ISSN 2077-0472. doi: 10.3390/agriculture14010057. URL https://www.mdpi.com/2077-0472/14/1/57.

Yu Fan Chen, Miao Liu, Michael Everett, and Jonathan P. How. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 285–292, 2017. doi: 10.1109/ICRA.2017.7989037.

Yuxiao Chen, Ugo Rosolia, and Aaron D Ames. Decentralized task and path planning for multi-robot systems. *IEEE Robotics and Automation Letters*, 6(3):4337–4344, 2021.

Howie Choset, Kevin M. Lynch, Seth Hutchinson, George A. Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. The MIT Press, 2005.

Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Amanda Coles, Andrew Coles, Maria Fox, and Derek Long. Forward-chaining partial-order planning. *Proceedings of the International Conference on Automated Planning and Scheduling*, 20:42–49, May 2010. ISSN 2334-0835. doi: 10.1609/icaps.v20i1.13403. URL http://dx.doi.org/10.1609/icaps.v20i1.13403.

Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

Ian Craig, Nicholas Woods, and Gary Dorr. A simple guide to predicting aircraft spray drift. *Crop Protection*, 17(6):475–482, 1998. doi: 10.1016/S0261-2194(98)00006-4.

Mariano Gonzalez de Soto, Luis Emmi, Manuel Perez-Ruiz, Juan Aguera, and Pablo Gonzalez de Santos. Autonomous systems for precise spraying–evaluation of a robotised patch sprayer. *Biosystems Engineering*, 146:165–182, 2016. ISSN 1537-5110. doi: https://doi.org/10.1016/j.biosystemseng.2015.12.018. URL https://www.sciencedirect.com/science/article/pii/S153751101500197X. Special Issue: Advances in Robotic Agriculture for Crops.

Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*, 2020.

Gabriel Dulac-Arnold, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, April 2021. ISSN 1573-0565. doi: 10.1007/s10994-021-05961-4. URL http://dx.doi.org/10.1007/s10994-021-05961-4.

Roger K. Dumbauld, James R. Bjorklund, and Sherry F. Saterlie. Computer models for predicting aircraft spray dispersion and deposition above and within forest canopies: User's manual for the FSCBG computer program. Technical Report 80-11, H. E. Cramer Company, Inc., prepared for the USDA Forest Service, 1980.

Chijioke C. Ekechi, Tarek Elfouly, Ali Alouani, and Tamer Khattab. A survey on uav control with multi-agent reinforcement learning. *Drones*, 9(7), 2025. ISSN 2504-446X. doi: 10.3390/drones9070484. URL https://www.mdpi.com/2504-446X/9/7/484.

Michael Everett, Yu Fan Chen, and Jonathan P. How. Motion planning among dynamic, decision-making agents with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3052–3059. IEEE Press, 2018. doi: 10.1109/IROS.2018.8593871. URL https://doi.org/10.1109/IROS.2018.8593871.

Tingxiang Fan, Pinxin Long, Wenxi Liu, and Jia Pan. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. *The International Journal of Robotics Research*, 39(7):856–892, 2020. doi: 10.1177/0278364920916531. URL https://doi.org/10.1177/0278364920916531.

Amir Farid, Jafar Roshanian, and Malek Mouhoub. Multiple aerial/ground vehicles coordinated spraying using reinforcement learning. *Engineering Applications of Artificial Intelligence*, 151:110686, 2025.

Enrique Fernandez-Gonzalez, Brian Williams, and Erez Karpas. Scottyactivity: Mixed discrete-continuous planning with convex optimization. *Journal of Artificial Intelligence Research*, 62: 579–664, July 2018. ISSN 1076-9757. doi: 10.1613/jair.1.11219. URL http://dx.doi.org/10.1613/jair.1.11219.

M. Fox and D. Long. Pddl2.1: An extension to pddl for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20:61–124, December 2003. ISSN 1076-9757. doi: 10.1613/jair.1129. URL http://dx.doi.org/10.1613/jair.1129.

Maria Fox and Derek Long. Modelling mixed discrete-continuous domains for planning. *Journal of Artificial Intelligence Research*, 27:235–297, 2006.

Georg Goldenits, Kevin Mallinger, Sebastian Raubitzek, and Thomas Neubauer. Current applications and potential future directions of reinforcement learning-based digital twins in agriculture. *Smart Agricultural Technology*, 8:100512, 2024. ISSN 2772-3755. doi: https://doi.org/10.1016/j.atech.2024.100512. URL https://www.sciencedirect.com/science/article/pii/S2772375524001175.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Ruihua Han, Shengduo Chen, Shuaijun Wang, Zeqing Zhang, Rui Gao, Qi Hao, and Jia Pan. Reinforcement learned distributed multi-robot navigation with reciprocal velocity obstacle shaped rewards. *IEEE Robotics and Automation Letters*, 7(3):5896–5903, 2022. doi: 10.1109/LRA.2022.3161699.

Ziyuan Hao, Xinze Li, Chao Meng, Wei Yang, and Minzan Li. Adaptive spraying decision system for plant protection unmanned aerial vehicle based on reinforcement learning. *International Journal of Agricultural and Biological Engineering*, 15(4):16–26, 2022. ISSN 1934-6344. doi: 10.25165/j.ijabe.20221504.6929. URL http://www.ijabe.net/en/article/doi/10.25165/j.ijabe.20221504.6929.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Jing Huang, Baihui Du, Youmin Zhang, Quan Quan, Ban Wang, and Lingxia Mu. A pesticide spraying mission allocation and path planning with multicopters. *IEEE Transactions on Aerospace and Electronic Systems*, 60(2):2277–2291, 2024. doi: 10.1109/TAES.2024.3355028.

Riashat Islam, Peter Henderson, Maziar Gomrokchi, and Doina Precup. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. *arXiv preprint arXiv:1708.04133*, 2017.

ISO. *ISO 22866:2005 Equipment for crop protection — Methods for field measurement of spray drift*. International Organization for Standardization, Geneva, Switzerland, 2005. URL https://www.iso.org/standard/35161.html.

Chanyoung Ju, Jeongeun Kim, Jaehwi Seol, and Hyoung Il Son. A review on multirobot systems in agriculture. *Computers and Electronics in Agriculture*, 202:107336, 2022. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2022.107336. URL https://www.sciencedirect.com/science/article/pii/S0168169922006457.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998. doi: 10.1016/S0004-3702(98)00023-X.

Mahsa Khosravi, Zhanhong Jiang, Joshua R. Waite, Sarah E. Jones, Hernan Torres Pacin, Arti Singh, Baskar Ganapathysubramanian, Asheesh Kumar Singh, and Soumik Sarkar. Optimizing navigation and chemical application in precision agriculture with deep reinforcement learning and conditional action tree. *Smart Agricultural Technology*, 12:101194, 2025. ISSN 2772-3755. doi: https://doi.org/10.1016/j.atech.2025.101194. URL https://www.sciencedirect.com/science/article/pii/S2772375525004253.

Peter E. Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*, volume 23 of *Applications of Mathematics*. Springer-Verlag, Berlin, 1992. doi: 10.1007/978-3-662-12616-5.

Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 4501–4510, 2020.

Arsenii Kuznetsov, Pavel Shvechikov, Alexander Grishin, and Dmitry Vetrov. Controlling overestimation bias with truncated mixture of continuous distributional quantile critics. In *International conference on machine learning*, pp. 5556–5566. PMLR, 2020.

Ratan Lal and Pavithra Prabhakar. Time-optimal multi-quadrotor trajectory planning for pesticide spraying. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7965–7971, 2021. doi: 10.1109/ICRA48506.2021.9561972.

Ratan Lal, Ajay Sharda, and Pavithra Prabhakar. Optimal multi-robot path planning for pesticide spraying in agricultural fields. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 5815–5820, 2017. doi: 10.1109/CDC.2017.8264538.

Duong Le and Erion Plaku. Cooperative multi-robot sampling-based motion planning with dynamics. *Proceedings of the International Conference on Automated Planning and Scheduling*, 27:513–521, June 2017. ISSN 2334-0835. doi: 10.1609/icaps.v27i1.13854. URL http://dx.doi.org/10.1609/icaps.v27i1.13854.

Duong Le and Erion Plaku. Multi-robot motion planning with dynamics via coordinated sampling-based expansion guided by multi-agent search. *IEEE Robotics and Automation Letters*, 4(2): 1868–1875, 2019. doi: 10.1109/LRA.2019.2898087.

Duong Le and Erion Plaku. Multi-robot motion planning with unlabeled goals for mobile robots with differential constraints. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7950–7956, 2021. doi: 10.1109/ICRA48506.2021.9560883.

Jiaoyang Li, Pavel Surynek, Ariel Felner, Hang Ma, T. K. Satish Kumar, and Sven Koenig. Multi-agent path finding for large agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7627–7634, July 2019. ISSN 2159-5399. doi: 10.1609/aaai.v33i01.33017627. URL http://dx.doi.org/10.1609/aaai.v33i01.33017627.

Yang Li, Yang Xu, Xinyu Xue, Xuemei Liu, and Xinghua Liu. Optimal spraying task assignment problem in crop protection with multi-uav systems and its order irrelevant enumeration solution. *Biosystems Engineering*, 214:177–192, 2022. ISSN 1537-5110. doi: https://doi.org/10.1016/j.biosystemseng.2021.12.018. URL https://www.sciencedirect.com/science/article/pii/S1537511021003263.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6382–6393, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning. *CoRR*, abs/1803.07055, 2018. URL http://arxiv.org/abs/1803.07055.

Martin, Perine, Grant, Abi-Akar, Henry, and Latheef. Spray deposition and drift as influenced by wind speed and spray nozzles from a remotely piloted aerial application system. *Drones*, 9(1):66, 2025. doi: 10.3390/drones9010066.

Belay Tizazu Mengistie, Ram L. Ray, and Ayodeji Iyanda. Environmental and human health impacts of agricultural pesticides on bipoc communities in the united states: A review from an environmental justice perspective. *International Journal of Environmental Research and Public Health*, 22(11), 2025. ISSN 1660-4601. doi: 10.3390/ijerph22111683. URL https://www.mdpi.com/1660-4601/22/11/1683.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937. PMLR, 2016.

Fazal E. Nasir, Muhammad Tufail, Muhammad Haris, Jamshed Iqbal, Said Khan, and Muhammad Tahir Khan. Precision agricultural robotic sprayer with real-time tobacco recognition and spraying system based on deep learning. *PLOS ONE*, 18(3):1–22, 03 2023. doi: 10.1371/journal.pone.0283801. URL https://doi.org/10.1371/journal.pone.0283801.

Zepeng Ning and Lihua Xie. A survey on multi-agent reinforcement learning and its application. *Journal of Automation and Intelligence*, 3(2):73–91, 2024. ISSN 2949-8554. doi: https://doi.org/10.1016/j.jai.2024.02.003. URL https://www.sciencedirect.com/science/article/pii/S2949855424000042.

Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems. Springer, 2016. doi: 10.1007/978-3-319-28929-8.

Michael Ostertag, Nikolay Atanasov, and Tajana Rosing. Trajectory planning and optimization for minimizing uncertainty in persistent monitoring applications. *Journal of Intelligent & Robotic Systems*, 106(1):2, Aug 2022. ISSN 1573-0409. doi: 10.1007/s10846-022-01676-3. URL https://doi.org/10.1007/s10846-022-01676-3.

Pygame community. Pygame. https://www.pygame.org/, 2000-2024. Version 2.1.0.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.

Mostafa Rahimi Azghadi, Alex Olsen, Jake Wood, Alzayat Saleh, Brendan Calvert, Terry Granshaw, Emilie Fillols, and Bronson Philippa. Precision robotic spot-spraying: Reducing herbicide use and enhancing environmental outcomes in sugarcane. *Computers and Electronics in Agriculture*, 235:110365, 2025. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2025.110365. URL https://www.sciencedirect.com/science/article/pii/S0168169925004715.

Tabish Rashid, Mikayel Samvelyan, Christian Schröder de Witt, Gregory Farquhar, Jakob N. Foerster, and Shimon Whiteson. QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, Proceedings of Machine Learning Research, pp. 4292–4301. PMLR, 2018. URL http://proceedings.mlr.press/v80/rashid18a.html.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

Daniele Reda, Tianxin Tao, and Michiel van de Panne. Learning to locomote: Understanding how environment design matters for deep reinforcement learning. In *Proceedings of the 13th ACM SIGGRAPH conference on motion, interaction and games*, pp. 1–10, 2020.

E. Rohmer, S. P. N. Singh, and M. Freese. Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013. www.coppeliarobotics.com.

John Schulman, Sergey Levine, Pieter Abbeel, Michael I Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, pp. 1889–1897. PMLR, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017. URL https://arxiv.org/abs/1707.06347.

Guni Sharon, Roni Stern, Ariel Felner, and Nathan R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66, 2015. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2014.11.006. URL https://www.sciencedirect.com/science/article/pii/S0004370214001386.

Chander Shekhar, Reetu Khosya, Kushal Thakur, Danish Mahajan, Rakesh Kumar, Sunil Kumar, and Amit Kumar Sharma. A systematic review of pesticide exposure, associated risks, and long-term human health impacts. *Toxicol. Rep.*, 13(101840):101840, December 2024.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, pp. 2085–2087, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2 edition, 2018.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Milton E. Teske, Steven L. Bird, David M. Esterly, Thomas B. Curbishley, Scott L. Ray, and Steven G. Perry. AgDRIFT: A model for estimating near-field spray drift from aerial applications. *Environmental Toxicology and Chemistry*, 21(3):659–671, 2002. doi: 10.1002/etc.5620210327.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 23–30. IEEE Press, 2017. doi: 10.1109/IROS.2017.8202133. URL https://doi.org/10.1109/IROS.2017.8202133.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

George E. Uhlenbeck and Leonard S. Ornstein. On the theory of the Brownian motion. *Physical Review*, 36(5):823–841, 1930.

Aruna Varanasi, P.V. Vara Prasad, and Mithila Jugulam. Chapter three - impact of climate change factors on weeds and herbicide efficacy. In Donald L. Sparks (ed.), *Advances in Agronomy*, volume 135 of *Advances in Agronomy*, pp. 107–146. Academic Press, 2016. doi: https://doi.org/10.1016/bs.agron.2015.09.002. URL https://www.sciencedirect.com/science/article/pii/S0065211315001297.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

Yang Xu, Xinyu Xue, Zhu Sun, Wei Gu, Longfei Cui, Yongkui Jin, and Yubin Lan. Joint path planning and scheduling for vehicle-assisted multiple unmanned aerial systems plant protection operation. *Computers and Electronics in Agriculture*, 200:107221, 2022. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2022.107221.

Yang Xu, Yuxing Han, Zhu Sun, Wei Gu, Yongkui Jin, Xinyu Xue, and Yubin Lan. Path planning optimization with multiple pesticide and power loading bases using several unmanned aerial systems on segmented agricultural fields. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(3):1882–1894, 2023. doi: 10.1109/TSMC.2022.3205695.

Yang Xu, Xinyu Xue, Zhu Sun, Longfei Cui, Chen Cai, Qingqing Zhou, Chen Chen, and Wei Gu. Path planning and scheduling of multiple unmanned ground vehicles for orchard plant protection operations. *Computers and Electronics in Agriculture*, 237:110615, 2025. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2025.110615. URL https://www.sciencedirect.com/science/article/pii/S0168169925007215.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

ZeroMQ. Zeromq: High-performance asynchronous messaging library. https://zeromq.org/, 2024. Last accessed on August 20, 2024.

Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *The World Wide Web Conference*, WWW '19, pp. 3620–3624, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3314139. URL https://doi.org/10.1145/3308558.3314139.

Wei Zhou, Mengmeng Li, and Varenyam Achal. A comprehensive review on environmental and human health impacts of chemical pesticide usage. *Emerging Contaminants*, 11(1):100410, 2025. ISSN 2405-6650. doi: https://doi.org/10.1016/j.emcon.2024.100410. URL https://www.sciencedirect.com/science/article/pii/S2405665024001112.

Karl J. Åström and Richard M. Murray. *Feedback Systems*. Princeton University Press, 2020.

A LIMITATIONS

Assumptions on the field. The field polygon, the locations of the infection sites and their initial intensities are assumed known at the start of an episode; only the robots' initial positions (Gaussian jitter around fixed nominal points, redrawn until no two robots overlap) and the initial wind (Section C.2) are randomized, and the infection map reaches the policy through a noisy observation channel rather than an onboard detector. A deployment would instead have to estimate the map online from perception, which turns the task into an active-sensing problem. The initial positions are drawn around fixed nominal points, whereas a real fleet would launch from a station at a fixed location; modeling that only narrows the initial state distribution and is therefore a simplification in the conservative direction.

Assumptions on the fleet. All robots are identical, share the same dynamics and carry the same spraying mechanism. Heterogeneous fleets, mid-episode failures and battery constraints are not modeled. The episode horizon T is long enough for every variation to be completed (Table 6), so the regime in which the flight time is the binding constraint — and the optimal policy must decide which sites to abandon — is left unexplored. The joint action space grows as $n(d+1)$ and the centralized observation as $(2d+1)n+m+d$, so the centralized formulation scales poorly with fleet size; we benchmark it up to $n=5$ in the main runs (and TQC up to $n=8$ in the extended benchmark, Section E.3). The decentralized formulation does not suffer this growth, since the neighbor budget ν fixes $\dim(\Omega_i)$ (Section C.6); we benchmark it, and the model-based controller, up to $n=8$.

Guarantees. Because the actuation, position and spray noises and the wind innovations are unbounded Gaussians, no policy can keep the robots inside the workspace or collision-free with probability one. The constraints of Eqs. 13–15 are therefore stated as chance constraints with tolerance δ, and the reward of Eq. 17 penalizes violations rather than forbidding them. The learned policies satisfy the constraints empirically at the rates reported in the terminal condition columns, but nothing here certifies them. Closing that gap needs either an explicit safety filter at execution time or post-hoc verification (for example, statistical model checking against a temporal-logic specification); both are outside the scope of this benchmark.

Uncertainty model. The deposition kernel of Eqs. 5–6 is a deliberately lightweight surrogate for the mechanistic drift models discussed in Section 4. Its parameters are calibrated (Section C.3) against a single published field model of UAV spray drift (Martin et al., 2025) rather than against AGDISP/AgDRIFT curves, which we could not obtain, or against raw ISO 22866 measurements. That reference is the authors' fitted mean model rather than raw sampler data, it comes from one site and one UAV, and it observes only the downwind tail ($d \geq 3.5$ m) of the deposit: the in-swath peak is constrained only through the functional form, the plume-elongation and wind-attenuation parameters σ_{dr} and η_w are not identified, and the reference tail is heavier than that of any Gaussian. The tank-draw coefficient k_β cannot be identified from deposition data at all and keeps its nominal value. The nominal wind of each variation (0.5–3.0 m/s, Table 6) is chosen within the range commonly recommended for spray application rather than measured at a site, and the wind correlation time τ_w is a modeling choice (Section C.2). We also do not characterize the maximum noise variance the task tolerates, for the wind or for any other channel, so the robustness envelope of the environment is unknown. Droplet evaporation, canopy interception and terrain are not modeled, and the wind is temporally correlated but spatially uniform across the field, whereas real gusts are correlated in space as well.

Dynamics. The robot model is a planar double integrator with linear drag ($k_w\,dt/W_i = 0.1$, a velocity time constant of 10 s), integrated by an explicit Euler–Maruyama step. This is simple by design: the formulation, the reward and both observation models are independent of the particular f_i, so richer dynamics — full quadrotor attitude control, ground vehicles, or non-holonomic constraints — can be substituted without touching anything else. We regard this as an extension point rather than a limitation of the formulation.

Experimental budget. To fit the study into a fixed compute budget, hyperparameters are tuned with 12 trials per algorithm (each trial trains 9 runs), and only at $n=3$; results at other team sizes use either default hyperparameters or hyperparameters transferred from $n=3$ without re-tuning

(Table 1 reports both; the scalability study, Section E.5, uses the transferred ones). The reward and observation ablations use 3 seeds per variation instead of 5, and the ablations, the observation gap and the scalability study cover the two decentralized learners and the model-based controller but, outside the extended benchmark of Section E.3, not the centralized learners.

B Extended Related Work

This appendix expands on the four themes introduced in Section 2, providing additional context, citations, and discussion of works that are relevant but space-constrained in the main paper.

B.1 Precision Spraying: Single-Robot Systems

The earliest targeted spraying systems coupled GPS-guided variable-rate applicators with prescription maps derived from multispectral or satellite imagery, enabling site-specific nozzle control without onboard perception (de Soto et al., 2016; Allmendinger et al., 2022). The shift toward real-time onboard detection brought two complementary advances: richer sensor modalities and tighter control. Nasir et al. (2023) developed a differential-drive ground robot with a YOLO-based detection pipeline for tobacco pest management, achieving 87.2% F1-score detection accuracy and up to 60% agrochemical reduction through a novel pressure-and-flow control mechanism that suppresses spray fluctuations. Rahimi Azghadi et al. (2025) demonstrated in 25 ha of field trials that a computer-vision spot sprayer reduces herbicide usage by approximately 35% while maintaining 97% of the weed control efficacy of broadcast spraying. Chang et al. (2024) proposed a four-wheeled robot that combines deep-learning weed detection with PID speed control and fuzzy-logic steering for row-crop navigation.

UAV-mounted platforms extend the reach of single-vehicle systems to large or irregular fields (Ju et al., 2022; Baltazar et al., 2021), but they typically operate under pre-planned, deterministic trajectories and are deployed one vehicle at a time. CNN-based and hardware-prototype spot sprayers further demonstrate commercial demand for precision application (Allmendinger et al., 2022; Baltazar et al., 2021; Ju et al., 2022). Across this body of work, a unifying limitation is the use of pre-programmed or rule-based controllers that lack closed-loop adaptability to changing conditions and do not consider multi-robot coordination.

B.2 Precision Spraying: Multi-Robot Coordination

Multi-robot spraying has been cast primarily as a combinatorial optimization problem. Lal et al. (2017); Lal & Prabhakar (2021) formulated the task as a constrained mixed-integer linear program and solved it with clustering-based decomposition, treating field geometry and infection locations as fixed deterministic inputs. Xu et al. (2023) developed a hybrid heuristic for multi-UAV path planning across segmented farmland with multiple refilling bases, minimizing total energy and deadhead flight. Xu et al. (2022) extended this to joint ground-vehicle–UAV scheduling, co-optimizing vehicle stops and UAV sorties over large-scale fields. Li et al. (2022) studied task assignment under capacity constraints via order-irrelevant enumeration. Huang et al. (2024) modeled multi-UAV spraying in hilly terrain as a multi-traveler problem and derived terrain-aware allocation strategies. Xu et al. (2025) addressed scheduling and path planning for multiple UGVs in orchard pest control with multiple supply bases.

These methods provide principled formulations and strong performance in structured settings. However, they share three limitations that motivate the present work: (i) they assume known and fixed field geometry and infection distributions; (ii) they operate under deterministic dynamics, producing plans that are either fully open-loop or require explicit replanning; and (iii) they do not explicitly model stochastic disturbances such as wind, actuation noise, or chemical deposition uncertainty. When deployed in realistic agricultural fields where these factors are unavoidable, the resulting compounding errors can significantly degrade performance.

B.3 Task Allocation and Classical Planning

PDDL (Fox & Long, 2003) and its extension PDDL+ (Fox & Long, 2006)—which supports continuous processes and events—provide expressive languages for specifying multi-robot planning problems with temporal and resource constraints. Planners built on these formalisms, including

POPF (Coles et al., 2010) and OPTIC (Benton et al., 2012), handle continuous numeric effects over time. ScottyActivity (Fernandez-Gonzalez et al., 2018) pushes further by addressing hybrid activity and trajectory planning for robot teams over long horizons, interleaving forward search with convex optimization over continuous state and control variables.

For multi-agent path finding (MAPF), Conflict-Based Search (Sharon et al., 2015) and its extensions (Li et al., 2019) provide optimal or bounded-suboptimal solutions in discrete and continuous-time settings. Sampling-based approaches with kinodynamic constraints (Le & Plaku, 2017; 2019; 2021; Bui et al., 2022) generate high-quality trajectories in continuous state-action spaces and have been extended to multi-robot problems with unlabeled goals and differential constraints. Despite their sophistication, all of these methods share a critical limitation in our setting: they assume deterministic or at most mildly uncertain dynamics. PDDL and PDDL+ model action effects exactly, without probabilistic transitions. ScottyActivity explicitly assumes first-order linear dynamics with no runtime actuation uncertainty. Sampling-based planners construct deterministic motion trees over precisely modeled dynamics (Le & Plaku, 2017; 2019); while they handle continuous spaces, they do not model stochastic wind disturbances, noisy spray deposition, or partial observability at runtime. CBS-based MAPF methods operate on discretized graphs with deterministic, known edge costs (Sharon et al., 2015). As a result, all of these methods produce plans that are susceptible to compounding errors when deployed in physically realistic agricultural environments with continuous stochastic perturbations.

Reinforcement learning sidesteps this limitation by optimizing closed-loop policies that directly maximize expected cumulative reward under the full distribution of environment dynamics, making it better suited to the stochastic, partially observable setting studied here.

B.4 Reinforcement Learning in Robotics and Agriculture

Agricultural robotics. RL has been applied to agricultural tasks including crop monitoring, harvesting, and navigation (Hao et al., 2022; Goldenits et al., 2024). For adaptive spraying, Hao et al. (2022) trained a UAV to select spray parameters from crop-condition observations, achieving pesticide reductions while treating navigation and coordination as out-of-scope. HAM-PPO (Khosravi et al., 2025) is the most closely related prior work: it uses a hierarchical PPO policy with conditional action masking to jointly learn navigation and spray decisions for a single robot, demonstrating that RL can integrate motion planning and chemical application within a unified framework. However, HAM-PPO's simulation environment is deliberately simplified and does not model wind disturbances, actuation noise, or stochastic spray deposition. Extending this framework to a stochastic multi-robot formulation remains an open problem that we address directly. Surveys of RL in agriculture and UAV systems (Goldenits et al., 2024; Ekechi et al., 2025) confirm that RL-based spraying is largely limited to single-agent settings with deterministic environments.

Multi-robot navigation. In broader robotics, RL has achieved strong performance in multi-robot navigation and collision avoidance (Chen et al., 2017; 2021; Everett et al., 2018; Fan et al., 2020; Han et al., 2022). These works develop decentralized policies for safe navigation among static or dynamic obstacles, but neither addresses spraying tasks nor models physical uncertainty sources such as wind or actuation noise. Real-world deployment challenges for RL are well documented: Dulac-Arnold et al. (2021) identifies partial observability, sensor noise, and stochastic dynamics as the principal barriers to reliable deployment, while Henderson et al. (2018) highlights sensitivity to environment design and reproducibility as systemic concerns. Both are particularly acute in precision agriculture.

Multi-agent RL. MARL provides scalable frameworks for decentralized coordination. Value decomposition methods—VDN (Sunehag et al., 2018) and QMIX (Rashid et al., 2018; 2020)—factorize joint value functions to make cooperative learning tractable. Actor-critic approaches—MADDPG (Lowe et al., 2017) and MAPPO (Yu et al., 2022)—achieve strong empirical performance in cooperative settings. These methods have been applied to large-scale coordination problems, but are typically evaluated on synthetic benchmarks with simplified dynamics, limiting direct applicability to physically grounded agricultural environments.

Stochasticity in robot learning. Handling environmental stochasticity is a recognized difficulty in robot RL (Dulac-Arnold et al., 2021; Henderson et al., 2018; Islam et al., 2017). Domain

randomization—randomly perturbing environment parameters across training episodes—is widely used to improve sim-to-real transfer (Akkaya et al., 2019; Tobin et al., 2017). Stochastic dynamics have been considered for single-robot trajectory planning under uncertainty (Ostertag et al., 2022), and stochastic games provide a formal framework for multi-agent settings (Ning & Xie, 2024). In practice, however, most cooperative RL environments remain deterministic or only partially stochastic. To our knowledge, no prior work jointly incorporates stochastic wind, actuation noise, spray-deposition uncertainty *and* observation noise inside a cooperative multi-robot precision spraying environment.

B.5 Environment Design and Benchmarking

The importance of carefully designed RL environments is well recognized (Reda et al., 2020; Cobbe et al., 2020; Dulac-Arnold et al., 2020; Sutton & Barto, 2018; Henderson et al., 2018). Standard benchmarks—Google Research Football (Kurach et al., 2020) and CityFlow (Zhang et al., 2019)—provide useful evaluation platforms but lack formal MDP specifications and are disconnected from physical robotics or agricultural tasks. The Gymnasium interface (Towers et al., 2024; Brockman et al., 2016) has become the de facto standard for environment development and algorithm comparison.

Physics-based simulators—MuJoCo (Todorov et al., 2012), PyBullet (Coumans & Bai, 2016–2021), and Isaac Gym (Makoviychuk et al., 2021)—enable realistic modeling of robot dynamics and contact interactions, and have underpinned sim-to-real transfer research (Tassa et al., 2018; Akkaya et al., 2019; Tobin et al., 2017). Many Gymnasium-based environments nonetheless rely on lightweight 2-D frameworks such as Pygame (Pygame community, 2000-2024) that omit real-time physics, limiting fidelity for outdoor robotic tasks.

Our environment is fully Gymnasium-compatible and built on the PyBullet (Coumans & Bai, 2016–2021) physics engine inside the CoppeliaSim (Rohmer et al., 2013) robotics simulator, combining continuous joint control of multiple robots (planar dynamics, deployed in a 3-D scene) with all four stochastic noise sources—wind, actuation, spray deposition, and observation—within a single environment. To our knowledge, no existing benchmark addresses this combination, and we argue it is essential for producing RL policies that can be deployed in realistic precision-agriculture settings. We further evaluate on-policy, off-policy, trust-region, distributional, actor–critic and value-factorization learners without modification, alongside a model-based controller, under a held-out evaluation protocol with confidence intervals, in line with the reproducibility standards called for by Henderson et al. (2018).

C Environment Implementation Details

This appendix fixes every quantity that Sections 4 and 5 leave symbolic, so that the environment is reproducible from the paper alone.

C.1 Parameter values

Table 4 lists the value of every symbol of Section 4. The workspace is planar ($d = 2$) and the integration step is $dt = 1$, which makes Eqs. 3–4 reduce to forward Euler while keeping the $\sqrt{dt}$ noise scaling explicit, so a finer grid can be used without re-tuning any noise level. The per-step physics (deposition, dynamics, out-of-field test and collision distance) is implemented as compiled kernels fed with standard-normal variates drawn from the environment's own seeded generator, so seeding semantics are those of a standard Gymnasium environment.

Five choices deserve comment.

The drag coefficient k_w sets the inertia. With $k_w = 0.1$ and $W_i = 1$, Eq. 3 keeps 90% of the previous velocity at every step, so a robot has momentum and must brake before a site. Full thrust along one axis reaches the terminal airspeed $k_a/k_w = 5\,\text{m/s}$, which equals $v_{\max}$, so the velocity bound is the physical top speed rather than an inactive safeguard. Because the drag acts on the airspeed $\mathbf{w}(t) - v_i(t)$, an unpowered robot relaxes toward the wind velocity with a time constant $W_i/k_w = 10\,\text{s}$: the wind is a persistent push that the policy must counter, not only a perturbation of the spray.

Collisions and the initial state. Each robot occupies a disc $\mathcal{A}_i$ of radius 1.0, so constraint (iii) is violated when two centers are closer than 2.0. The nominal launch points of every variation are at least 2.48 apart, for all eight launch points; the per-episode jitter $\mathcal{N}(0, 0.5^2)$ is redrawn until no two discs overlap, so b_0 is supported on collision-free states. Without this, up to 20% of episodes on the most crowded five-robot configuration (variation 8, $n = 5$) would end in a collision before the first action, contradicting the feasibility assumption of Section 4.

The spray coefficient k_β sets the effective footprint. The rescaling $\chi_i(t)$ of Eq. 6 caps a robot's total deposition per step at $k_\beta \beta_i^{\text{eff}} dt$. With $k_\beta = 0.1$ and the calibrated kernel ($g_w = 1$, since $\sigma_{\text{dr}} = \eta_w = 0$), that cap binds whenever the Gaussian kernel exceeds 0.1, i.e. within $\sqrt{2 \ln 10}\, \beta_s \approx 2.15\, \beta_s \approx 7.7$ m of the advected release point; beyond that radius the deposition falls off as the kernel does. The task is therefore well posed: with three robots the theoretical minimum completion time is $\lceil \sum_j \ell_j(0)/(n k_\beta dt) \rceil = 12$ steps (11 at the 90% threshold), against the 17.6 (MAPPO) and 22.7 (VDN) steps that the tuned policies need at the 90% completion threshold (Table 22).

The spray noise σ_s must stay small. Eq. 6 adds ϵ_s to *every* robot–site pair with no distance cutoff and then clips at zero, so the noise is one-sided: its expected contribution is $\sigma_s \sqrt{dt/2\pi}$ per site regardless of range. A large σ_s would let a robot reduce infection anywhere in the field and make the deposition kernel irrelevant. With $\sigma_s = 2 \times 10^{-3}$ and $m = 6$ sites the range-independent contribution is $m \sigma_s \sqrt{dt/2\pi} \approx 4.8 \times 10^{-3}$ per step, under 5% of the on-target cap $k_\beta \beta_i^{\text{eff}} dt$. This is a property of the model as written, not of the implementation: any σ_s large enough to matter as noise also makes range irrelevant.

The wind truncation bound. $v_w^{\max}$ is set equal to the robot speed limit $v_{\max}$: the wind may be as strong as the fastest commanded motion but no stronger. The nominal winds used here are at most 3.0 m/s with a stationary standard deviation $\sigma_w = 0.2$ m/s, so the upper bound is inactive in practice and the reflection only enforces $v_w \geq 0$, which is its stated purpose.

Terminal reward. Eq. 19 is implemented with a single magnitude $r_L = 2000$ for both completion and collision. It satisfies the requirement stated after Eq. 19: the largest cumulative spray reward an episode can collect is $r_s \sum_j \ell_j(0) = 500 \times 3.4 = 1700$, identical for every variation, and r_L exceeds it, so reaching or failing the terminal state outweighs any amount of partial spraying. The implementation asserts this bound. The same quantity defines the success threshold of Section 6: an episode that does not complete the task collects no terminal bonus, a non-positive path reward and a spray reward below $r_s(1 - \ell_\epsilon / \sum_j \ell_j(0)) \sum_j \ell_j(0)$, i.e. below 1700 at full completion and 1530 at the 90% threshold of the ablations, so a return above that value certifies completion.

One consequence is worth stating explicitly. Because the stage cost charges $\lambda_3 \sum_j \ell_j(t) + \lambda_4 \approx 10.5$ per step while infection remains, an episode that does nothing for the full horizon accrues $\approx -10{,}500$, and one whose robots drift out of the field accrues a further $r_s \lambda_b = 50$ per robot and step, whereas one that collides immediately accrues only ≈ -2000. Colliding is therefore *cheaper* than idling: the collision penalty bounds the damage of a failure but does not make failure the worst outcome, which is why r_{spray} and the terminal bonus have to carry the bulk of the signal.

Reward scale and the normalization of ℓ. Section 4 defines $\ell_j(t) \in [0, 1]$, so the raw intensities of each variation are divided by the largest one (5 throughout, see Table 6). The weights r_s and λ_3 absorb that normalizer exactly ($100 \to 500$ and $0.5 \to 2.5$), so episodic returns are on the same numerical scale as an implementation that keeps the raw intensities.

C.2 Wind process

By default the wind follows Eqs. 1–2 with a correlation time of $\tau_w = 20$ steps (20 s at $dt = 1$), so $\phi = e^{-1/20} \approx 0.951$ and a gust persists for roughly τ_w steps before it decorrelates. τ_w is a modeling choice rather than a measurement; the wind ablation brackets it against the uncorrelated limit. Both components are driven by independent innovations, the direction is not wrapped (its standard deviation of 5° keeps it far from $\pm 180^\circ$ of the nominal value), and the magnitude is reflected at 0 and $v_w^{\max}$, which keeps the process inside the bounds without piling probability mass on them.

The first draw of an episode comes from the stationary marginal, so the marginal distribution of $\mathbf{w}(t)$ is the same at every step and matches that of the independent model ($\phi = 0$), in which

Table 4: Environment parameter values. Symbols follow Section 4. Lengths are in field units (interpreted as meters), times in seconds. Spray-kernel parameters marked "cal." are fitted in Section C.3; the uncalibrated nominal value is given in parentheses.

Symbol	Meaning	Value	Where
Workspace and integration			
d	workspace dimension	2	Sec. 4
dt	integration step	1.0	Eq. 3
T	episode horizon (truncation)	1000 steps	Eq. 12
γ	discount factor	0.99	Sec. 5
Robot dynamics			
W_i	mass	1.0	Eq. 3
k_a	thrust coefficient	0.5	Eq. 3
k_w	aerodynamic drag coefficient	0.1	Eq. 3
$v_{\min}, v_{\max}$	velocity bounds	$-5.0,\ +5.0$	Eq. 3
$\mathcal{A}_i$	collision radius (disc)	1.0	Eq. 15
—	initial position jitter (per episode)	$\mathcal{N}(0, 0.5^2)$, collision-free	b_0
Wind			
$\bar{v}_w, \bar{\theta}_w$	nominal wind magnitude / direction	per variation (Table 6)	Eqs. 1–2
$v_w^{\max}$	magnitude bound	5.0 ($= v_{\max}$)	Sec. 4
σ_w	wind-magnitude stationary std.	0.20	Eq. 1
σ_θ	wind-direction stationary std.	5.0°	Eq. 2
τ_w	wind correlation time	20 steps ($\phi = e^{-1/20} \approx 0.951$); $\phi = 0$ in the i.i.d. ablation	Eqs. 1–2
Spray deposition			
β_s	still-air dispersion	3.59 cal. (2.5)	Eq. 5
$\bar{t}_w$	mean droplet fall time	0.80 cal. (1.0)	Eq. 6
σ_{dr}	fall-time standard deviation	0 cal., not identified (1.0)	Eq. 5
η_w	wind attenuation coefficient	0 cal., not identified (0.1)	Eq. 6
k_β	spray (tank-draw) coefficient	0.1 (not identifiable)	Eq. 8
ℓ_ϵ	completion threshold	0.01 (main), $0.1 \sum_j \ell_j(0)$ (ablations)	Eq. 13
Noise (full uncertainty mode; see Table 25)			
σ_a	actuation noise	0.10	Eq. 3
σ_p	position process noise	0.05	Eq. 4
σ_s	spray-deposition noise	2×10^{-3}	Eq. 6
σ_{obs}	state observation noise	0.01	Eq. 16
$\sigma_{w,obs}$	anemometer noise	0.02	Eq. 16
Reward (Eqs. 17–20)			
r_s	spray reward scale	500	Eq. 18
r_L	terminal magnitude	2000	Eq. 19
λ_1	energy weight	0.1	Eq. 20
λ_2	movement weight	0.3	Eq. 20
λ_3	infection weight	2.5	Eq. 20
λ_4	step weight	2.0	Eq. 20
λ_b	out-of-field weight	0.1	Eq. 20
Decentralized execution (Eq. 21)			
$\rho_{\mathcal{M}}$	onboard sensing radius	20.0	Sec. 5
ν	neighbor budget	4; also $n-1$ in the scalability study ($n > 5$)	Sec. 5

$v_w(t) \sim \mathcal{N}_{[0, v_w^{\max}]}(\bar{v}_w, \sigma_w^2)$ is redrawn by rejection at every step, up to the handling of the rarely reached bounds (reflection versus rejection). Only the temporal correlation differs between the two models, which is what the wind ablation of Table 27 isolates. The wind is sampled at the end of every step and in the reset, so the wind an agent observes in o_t is the one that acts during the transition it is about to take.

C.3 Spray-kernel calibration

Reference data. AGDISP and AgDRIFT are distributed as Windows GUI programs on request, and the tabulated AgDRIFT curves that are publicly available describe crewed aircraft over multi-swath fields at a single wind speed rather than a single UAV release across wind speeds. We therefore calibrate against the published UAV field study of Martin et al. (2025): an 8-rotor UAV spraying at 3 m height and 3 m/s ground speed, perpendicular to the wind, over a line of ground samplers, with 23 valid runs following the ASABE S561.1 protocol at 2-m wind speeds of 0.4–3.2 m/s. The authors fit a linear mixed-effects model on the cube-root scale; for the fine nozzle its fixed effects give the

Table 5: Spray-kernel calibration against the fine-nozzle deposition model of Martin et al. (2025) (108 points, least squares on the mass scale, free amplitude). R^2 of the uncalibrated kernel is computed with its amplitude fitted.

Parameter	**Uncalibrated**	**Calibrated ($\pm$ s.e.)**
β_s still-air dispersion [m]	2.5	3.59 ± 0.18
$\bar{t}_w$ mean fall time [s]	1.0	0.80 ± 0.32
σ_{dr} fall-time std. [s]	1.0	0 (not identified)
η_w wind attenuation [s/m]	0.1	0 (not identified)
R^2 vs. reference (mass scale)	0.87	0.94

deposition

$$D(d, W) = \big(4.94 - 1.28 \ln d + 0.82\, W - 0.30\, W \ln d\big)^3 \quad [\mu\mathrm{g/cm}^2] \tag{22}$$

at downwind distance d from the flight line and 2-m wind speed W. We evaluate Eq. 22 on the measured distance grid $d \in \{3.5, 4.5, \ldots, 20.5\}$ m for $W \in \{0.5, 1.0, \ldots, 3.0\}$ m/s, the nominal wind band of our variations, giving 108 reference points.

Fit. For a release at the origin under a wind of speed v along $+x$, the kernel of Eqs. 5–6 deposits

$$D_\theta(x, y; v) = A\, g_w(v) \exp\Big(- \tfrac{(x - \bar{t}_w v)^2}{2(\beta_s^2 + \sigma_{\mathrm{dr}}^2 v^2)} - \tfrac{y^2}{2\beta_s^2} \Big), \tag{23}$$

with g_w as in Section 4. Because the UAV flies perpendicular to the wind, the samplers see a line source: integrating the kernel over the crosswind axis multiplies it by the constant $\sqrt{2\pi}\,\beta_s$, which the free amplitude A (shared by all wind speeds) absorbs, so the center-line profile $D_\theta(x, 0; v)$ applies. We fit $(\beta_s, \bar{t}_w, \sigma_{\mathrm{dr}}, \eta_w, A)$ by bounded least squares on the linear (deposited-mass) scale: the environment uses the kernel to decide how much infection each site loses, which is governed by the near field where most of the mass lands. A fit on the authors' cube-root scale spends the kernel's width on the far tail and lowers R^2 on the mass scale from 0.94 to 0.81.

Result. Table 5 and Fig. 2 give the result. The still-air dispersion and the mean fall time are identified; σ_{dr} and η_w go to their lower bound of zero with very large standard errors, i.e. these data do not identify them, so the calibrated kernel is an isotropic Gaussian advected by $\bar{t}_w \mathbf{w}$. The reference tail decays roughly as $d^{-3.8}$, which no Gaussian reproduces, so the fit is a compromise that favors the near field (Fig. 2). k_β, a tank-draw rate, is not identifiable from deposition data and keeps its nominal value. Every training, tuning, evaluation, observation-gap and model-based-controller run uses the calibrated parameters, and each run records them in its configuration file. The fitting script accepts any other reference in the same tabular format (AGDISP/AgDRIFT exports or ISO 22866 measurements), and recovers known parameters from synthetic curves in a self-test.

C.4 Environment variations

We use ten randomized environment variations, listed in Table 6. Each is a convex or non-convex polygon of 4–8 vertices with area between 432 and 4420 field units^2, carrying $m = 6$ infection sites whose initial intensities are a permutation of $\{1, 2, 2, 3, 4, 5\}$ (normalized: $\{0.2, 0.4, 0.4, 0.6, 0.8, 1.0\}$, summing to $\sum_j \ell_j(0) = 3.4$), eight nominal launch positions with tank capacities in $\{7, \ldots, 12\}$, and a nominal wind. The first five launch positions of each variation are those of the original benchmark; positions six to eight were added for the scalability study, at least 3 from the others and at least 2 inside the field, and their capacities repeat those of robots one to three.

A variation is instantiated for a given team size n by taking the first n of its launch positions and capacities; the field, the infection sites, their intensities and the nominal wind are unchanged. The 10 randomized variations are thus shared across $n \in \{2, \ldots, 5\}$ in Table 1, and variations 1, 5 and 9 across $n \in \{2, \ldots, 8\}$ in Table 28, which makes the columns of each table a controlled comparison of team size rather than of map difficulty. Holding m, $\sum_j \ell_j(0)$ and the intensity multiset fixed across variations means the achievable return is comparable across maps, so the only thing that changes is the geometry the policy has to cover.

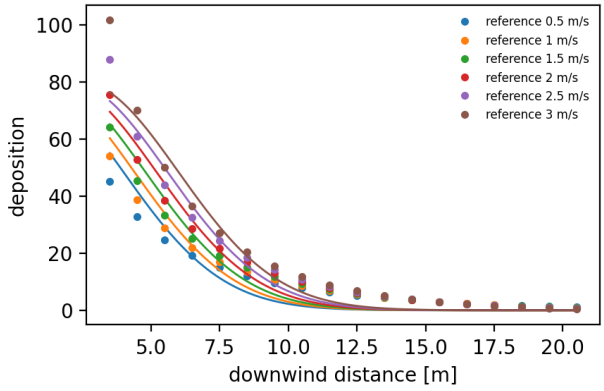


Figure 2: Calibrated kernel (lines) against the reference deposition of Martin et al. (2025) (dots, μg/cm^2) as a function of downwind distance, for wind speeds of 0.5–3.0 m/s.

Table 6: The ten randomized environment variations (`exp_sets/stochastic_envs_v2.json`). Every variation carries $m = 6$ infection sites with the same intensity multiset $\{1, 2, 2, 3, 4, 5\}$, hence the same normalized total $\sum_j \ell_j(0) = 3.4$. "Capacities" lists c_i^{init} for the eight nominal launch positions, of which the first n are used at team size n. Nominal wind is $(\bar{v}_w, \bar{\theta}_w)$ in m/s and degrees.

Variation	**Field vertices**	**Field area**	**Capacities** c_i^{init}	**Wind** $(\bar{v}_w, \bar{\theta}_w)$
1	5	3298.0	$10, 12, 8, 11, 9, 10, 12, 8$	$1.0, 0°$
2	5	4420.0	$9, 11, 10, 12, 8, 9, 11, 10$	$2.0, 45°$
3	4	432.0	$8, 10, 9, 11, 7, 8, 10, 9$	$0.5, 90°$
4	5	1500.0	$11, 9, 10, 12, 8, 11, 9, 10$	$1.5, 180°$
5	7	3640.0	$12, 10, 9, 11, 8, 12, 10, 9$	$2.5, 225°$
6	6	4256.0	$9, 8, 11, 10, 12, 9, 8, 11$	$1.0, 270°$
7	4	1030.0	$10, 9, 12, 11, 8, 10, 9, 12$	$3.0, 315°$
8	4	2648.0	$8, 10, 11, 9, 12, 8, 10, 11$	$2.0, 135°$
9	8	2599.5	$12, 11, 10, 9, 8, 12, 11, 10$	$1.5, 60°$
10	4	3252.0	$11, 9, 12, 10, 8, 11, 9, 12$	$2.5, 300°$

C.5 Centralized observation layout

The observation of Eq. 16 is laid out per robot, as the equation is written: the block $(p_i, \tilde{c}_i, \tilde{v}_i)$ of $2d + 1 = 5$ entries is repeated for $i = 1, \ldots, n$, followed by the m infection levels and the d wind components, giving $5n + m + d$. Absolute positions enter unnormalized, in field units; capacities and velocities enter normalized as defined after Eq. 16; infection levels are already in $[0, 1]$. The observation noise ϵ_o is applied with variance σ_{obs}^2 on the first $(2d + 1)n + m$ entries and $\sigma_{w,obs}^2$ on the last d.

The wind entry reports $\mathbf{w}(t)$, the wind that acts during the transition the agent is about to take, not the one that acted during the previous step (Section C.2). This keeps the observation a noisy function of s_t, as Eq. 16 requires, and gives the anemometer channel direct information about the next transition.

The four observation ablations of Table 23 drop whole blocks of this vector; Table 7 gives the resulting dimensions.

C.6 Decentralized observation layout

The local observation of Eq. 21 is assembled as $(p_i, \tilde{c}_i, \tilde{v}_i)$, the m infection levels, the d wind components, and then ν neighbor slots of $2d + 1 = 5$ entries each, giving $d_i = (2d + 1) + m + d + \nu(2d + 1) = 13 + 5\nu$ for $m = 6$, i.e. 33 at the default $\nu = 4$.

The neighbor set $\mathcal{K}_i(t)$ is built by sorting the other robots by distance and taking those within $\rho_{\mathcal{M}}$, up to ν of them; unused slots are filled with zeros, so $\mu_{i,k} = 0$ marks an empty slot and the policy can

Table 7: Observation dimensions under the four ablation conditions, for $d = 2$ and $m = 6$. Centralized: Eq. 16. Decentralized: Eq. 21, which additionally carries ν neighbor slots (of 3 entries under "Pos. only", which drops the relative velocity, and 5 otherwise).

Condition	**Blocks retained**	**Centralized** $\dim(\Omega)$		**Decentralized** $\dim(\Omega_i)$	
		general	n=3	general	ν=4
Full	positions, capacity+velocity, infection, wind	$5n + m + 2$	23	$13 + 5\nu$	33
No pos.	capacity+velocity, infection, wind	$3n + m + 2$	17	$11 + 5\nu$	31
No infect.	positions, capacity+velocity, wind	$5n + 2$	17	$7 + 5\nu$	27
Pos. only	positions	$2n$	6	$2 + 3\nu$	14

tell "no neighbor" from "a neighbor at zero relative displacement". Because ν is a fixed budget rather than $n - 1$, $\dim(\Omega_i) = 33$ for every team size in Table 1, and the same network architecture transfers across team sizes unchanged. $\nu = 4$ is the largest number of neighbors any robot can have at $n = 5$, so no information is discarded for $n \leq 5$. For $n \in \{6, 7, 8\}$ a robot sees only its four nearest visible teammates; the scalability study therefore compares $\nu = 4$ with $\nu = n - 1$ ($\dim(\Omega_i) = 38, 43, 48$), which sees every teammate within $\rho_{\mathcal{M}}$ at the cost of an input that grows with n.

Consistent with the diagonal $\mathbf{\Sigma}_{obs,i}$ of Section 5, the relative position and relative velocity entries carry noise of standard deviation σ_{obs}, the wind entries carry $\sigma_{w,obs}$, and the binary mask entries $\mu_{i,k}$ carry none.

Centralized critic. Under CTDE, MAPPO's critic receives the centralized observation of Eq. 16 — precisely the vector the single-agent baselines act on — while the actors condition only on local observations. VDN has no separate critic: its joint value is the sum of per-agent utilities. The critic therefore sees no information that the centralized POMDP does not already expose, and the actors strictly less, so any gap between the centralized and decentralized rows of Table 1 is attributable to the factored *policy* rather than to a change of the underlying task (S, $\mathcal{T}$, R, γ).

C.7 Action space and the discretization used by VDN

The per-robot action is $u_i = \langle \alpha_i, \beta_i \rangle$ with $\alpha_i \in [-1, 1]^d$ and $\beta_i \in [0, 1]$, so the joint action space is a $3n$-dimensional box for the centralized formulation and a 3-dimensional box per agent for the decentralized one. The MARL learners emit per-agent actions in $[-1, 1]^3$, which are affinely rescaled so that $\beta_i \in [0, 1]$ before they reach the environment.

VDN is defined for discrete action sets — it takes an argmax over per-agent utilities — so it acts on a factored grid that is decoded to a continuous action before it reaches the environment. The default grid is $5 \times 5 \times 3$: $\alpha_{i,x}$ and $\alpha_{i,y}$ each on $\{-1, -0.5, 0, 0.5, 1\}$ and β_i on $\{0, 0.5, 1\}$, giving 75 actions per agent. The number of thrust bins is itself a tuned hyperparameter ($\{3, 5, 7\}$, i.e. 27, 75 or 147 actions; Table 9). The discretization is a property of the algorithm, not of the environment: the dynamics, reward and termination conditions are unchanged, so VDN returns remain comparable to the other rows of Table 1, modulo the coarser policy class.

C.8 Input and reward conditioning of the learners

Observation normalization. Before an observation reaches any learned network — single-agent or multi-agent, actor or critic — it passes through a fixed affine transform: absolute and relative positions are centered on the field and divided by its half-extent, and the wind is divided by $v_w^{\max}$. Capacities, velocities and infection levels are already $O(1)$ by construction and are left alone. The constants come from the static field geometry, so the transform is deterministic, identical at training and evaluation time and carries no learned state. Without it, world-frame positions of magnitude $\approx 10^2$ saturate the $\tanh$ layers of the 64-unit networks at initialization, and in the MARL learners the per-agent one-hot index cannot break the symmetry between agents. This is a representation choice on the learner side; the environment, its reward, its dynamics and its termination conditions are untouched, so returns stay comparable across all methods. The model-based controller reads the unnormalized observation.

Reward scaling. Undiscounted returns reach $O(10^3$–$10^4)$, far outside the range that default value heads are built for, and bootstrapped Q-learning at that scale is badly conditioned. The single-agent learners are therefore trained on the reward multiplied by 0.01 (on the training environments only, so checkpoint selection and every evaluation use the raw reward), and VDN multiplies the reward by 0.01 when storing it in its replay buffer. Scaling the reward by a positive constant is an order-preserving transform of the value function: $\arg\max_a Q(s,a)$, and hence the learned policy, is unchanged. MAPPO does not use it — it normalizes value targets with a running estimator instead. Every number reported in this paper is the raw, unscaled environment reward, so all rows of Table 1 remain directly comparable.

D Model-Based Controller

The model-based controller is a hand-designed controller that uses the known dynamics and spray model. It is not trained, and it is scored with exactly the protocol of the RL runs: the same (variation, seed) cells and the same held-out episode seeds.

Control law. At every step the controller (i) groups the sites that are not yet cleared ($\ell_j > 10^{-3}$) into clusters, a site joining a cluster when it lies within `cluster_radius` of one of its members; (ii) assigns robots, in order of their distance to the nearest uncleared site, each to the nearest free cluster, and within it targets the site that maximises $\ell_j - 0.02\,\|z_j - p_i\|$; robots left without a cluster hold position; (iii) aims upwind of the target, at $z_j - \bar{t}_w \mathbf{w}(t)$, so that the advected release point $\tilde{p}_i$ of Eq. 6 lands on the site; (iv) sets a desired velocity towards the aim point with speed $\min(\texttt{v_cmd}, \texttt{gain}\cdot\text{dist})$, adds a pairwise repulsion $3\,((\texttt{sep}-r)/\texttt{sep})^2$ along the separation vector of every teammate closer than `sep`, and adds a unit pull towards the centroid of the launch points when $p_i + v_i\,dt$ would leave the field; (v) inverts Eq. 3 to obtain the thrust that tracks this velocity against drag and wind, $\alpha_i = \text{clip}\left((W_i(v_i^{\text{des}} - v_i)/dt - k_w(\mathbf{w} - v_i))/k_a, -1, 1\right)$; and (vi) sprays at full rate ($\beta_i = 1$) within `spray_radius` of its aim point and not at all otherwise. Its five parameters are listed in Table 8.

Observer. The controller acts on the unnormalized observation o_t of Eq. 16, including its sensor noise, the observation ablations and the observation-gap perturbations, not on the true state. Its model knowledge is static: field geometry, site locations, launch points, initial infection levels and tank capacities, the dynamics and spray-kernel parameters, and the variation's nominal wind. A small observer supplies what the observation does not carry. The noisy infection map is smoothed exponentially (weight 0.2 on a new reading), since the sensor noise $\sigma_{obs} = 0.01$ is ten times the controller's "site cleared" level; without the map (No infect., Pos. only) the levels are predicted by running the noise-free spray kernel on the controller's own spray commands; without positions (No pos.) it dead-reckons from the launch points with Eq. 4; without velocities (Pos. only) it differentiates the observed positions and propagates the estimate one step through Eq. 3 with its last command; and without the wind (Pos. only) it uses the nominal wind.

Tuning. The five parameters are tuned by random search with the objective of the RL tuning protocol (Section E.1): at $n = 3$, on the tuning variations $\{1, 4, 7\}$ and tuning seeds $\{7, 77, 777\}$, each configuration is scored by the median over these nine cells of the IQM of return over 20 held-out episodes. The search evaluates the untuned default configuration and 40 configurations sampled uniformly from the ranges of Table 8, and keeps the best. Table 1 reports the untuned controller in its default columns and the tuned one in its tuned columns, and every later study uses the tuned controller.

Feasibility check. Section 4 assumes that every instance is feasible. We check this empirically, before any learning, with the untuned controller given access to the true state rather than the observation: 20 episodes per (variation, team size) cell for the ten variations and $n \in \{2, \ldots, 8\}$, under the full noise model and the calibrated kernel. Over the 70 cells, the controller completes 72.8% of the episodes, and all remaining episodes end in a collision; none reaches the step limit. Completion falls with the team size, from 93.5% and 95.0% at $n \in \{2, 3\}$ to 82.0% at $n = 5$ and 31.5% at $n = 8$. At least one episode is completed in 68 of the 70 cells. The two exceptions are variation 2 at $n = 6$ and $n = 8$, where every episode ends in a collision. Because the failures are collisions of this simple repulsion rule rather than incomplete coverage, the check shows that every cell of Table 1 ($n \leq 5$)

Table 8: Parameters of the model-based controller: untuned default, random-search range (uniform) and selected value.

Parameter	Role	Default	Range	Tuned
`v_cmd`	maximum commanded speed [m/s]	2.0	$[1.0, 5.0]$	4.83
`gain`	speed gain on the distance to the aim [1/s]	0.4	$[0.1, 1.5]$	0.308
`sep`	repulsion radius [m]	7.0	$[3.0, 10.0]$	9.81
`cluster_radius`	site clustering radius [m]	6.0	$[2.0, 10.0]$	9.12
`spray_radius`	spraying radius around the aim [m]	3.0	$[1.0, 5.0]$	4.29
Tuning objective (median IQM)		3148		3338

can be completed, and for $n \geq 6$ every cell except those two. This is empirical support for, not a proof of, the chance-constrained feasibility assumed in Section 4, which requires completion with probability above $1 - \delta$.

E Additional Experimental Details

This appendix provides full experimental details and extended analyses that support the findings summarized in Section 6 of the main paper.

E.1 Experimental Setup Details

Algorithms. On the centralized POMDP, the main paper evaluates two on-policy algorithms, PPO (Schulman et al., 2017) and TRPO (Schulman et al., 2015). The off-policy TQC (Kuznetsov et al., 2020) is run with default hyperparameters under the same protocol and reported in this appendix only (Section E.2); a tuned TQC is evaluated in the extended benchmark (Section E.3). All three use Stable-Baselines3 and its contrib package (Raffin et al., 2021). PPO uses a clipped surrogate objective and TRPO a hard KL trust region; TQC extends SAC (Haarnoja et al., 2018) with an ensemble of distributional critics whose top quantiles are truncated to control overestimation.

On the Dec-POMDP we evaluate two cooperative MARL baselines under centralized training with decentralized execution, one from each of the two main families: MAPPO (Yu et al., 2022), an actor–critic method with a shared stochastic Gaussian actor and a centralized critic, and VDN (Sunehag et al., 2018), which factorizes the joint action value additively into per-agent utilities (double Q-learning, hard target updates) and acts on the discretized action set of Section C.7. Both share parameters across the homogeneous robots and append a per-agent one-hot index to the actor and critic inputs, which is what breaks the symmetry that parameter sharing would otherwise impose. All learned policies are feedforward networks. The model-based controller of Section D completes the set of methods.

Environment variations. We use the ten environment variations of Table 6, each instantiated at $N \in \{2, 3, 4, 5\}$ for Table 1, giving 40 (variation, team size) configurations, so results are not confounded by a single map layout. The scalability study adds $N \in \{6, 7, 8\}$ on variations 1, 5 and 9. Every run uses the calibrated spray kernel and the correlated wind model unless the row states otherwise.

Training protocol. Every algorithm–variation–team-size cell is trained with 5 independent random seeds $\{0, 42, 123, 2024, 9999\}$ for 2×10^6 environment steps, collected through 8 parallel environments for every learner. The budget is the same for the centralized and the decentralized baselines, so the centralized and decentralized rows of Table 1 see the same number of interactions with the task and any gap between them reflects the execution scheme rather than the amount of experience. Note that a decentralized run consumes N *agent* transitions per environment step, so the MARL baselines process more transitions per unit of environment interaction than the centralized ones; parity is on the axis the environment defines, not on gradient steps. If a run's networks diverge to non-finite values, training stops and the run is scored on the best checkpoint saved before the divergence; the step at which this happens is recorded with the run.

Compute. All training, tuning and evaluation runs are executed on the CPU nodes of an HPC cluster managed by SLURM, with no GPU. Every run is single-threaded and is allocated one core, 4 GB of memory and a 24 h limit; a tuning trial trains its nine runs in parallel on nine cores (18 GB, 48 h limit). The whole plan — including the calibration, the feasibility check, the model-based controller and the analysis — is generated from a single experiment description, which also drives a one-machine scheduler, so both routes run identical commands. Simulation and inference timing was measured on a local workstation with an AMD Ryzen Threadripper 1950X (16-core) CPU, 64 GB RAM and an NVIDIA GeForce RTX 2080 GPU.

Evaluation protocol. Checkpoint selection and scoring are separated. During training, each run is evaluated every 10^4 environment steps over 5 episodes with no exploration noise, and the checkpoint with the highest mean return is kept. When training ends, that checkpoint is re-evaluated on 50 held-out episodes whose seeds ($\geq 10^7$) never occur in training or checkpoint selection. The held-out episode seeds depend only on the (variation, seed) cell, so every method — including the model-based controller — faces the same initial conditions and noise streams on the same cell, and every comparison between methods is paired. The same episodes are replayed under every cross-evaluation condition of the ablations; the observation-gap study draws its own, independent set of 100 episodes per run. A run's score is its mean held-out return. Because the selected checkpoint is the best of many 5-episode evaluations, its selection-phase score is optimistic; we report the selection-phase IQM next to the held-out IQM (Table 14), and the difference quantifies that optimism.

Statistics. Each table cell is the interquartile mean (IQM) (Agarwal et al., 2021) of the run scores of the cell (5 seeds $\times$ 10 variations $= 50$ runs in Table 1); the IQM is more robust to outlier runs than the arithmetic mean. Confidence intervals are 95% stratified-bootstrap percentile intervals on the IQM with 5000 resamples, variations as strata and runs resampled within them, following Agarwal et al. (2021). A † marks an IQM above the success threshold of Section C.1 (1700 at full completion, 1530 at 90%). Paired comparisons in the observation-gap study use two-sided Wilcoxon signed-rank tests (Wilcoxon, 1945) over runs.

Hyperparameter settings. We run the main experiments under two regimes. *Default* hyperparameters are the Stable-Baselines3 / sb3-contrib defaults for the centralized baselines (with the input and reward conditioning of Section C.8 and 8 parallel environments) and the values of Table 12 for the MARL baselines. *Tuned* hyperparameters are found by Optuna (Akiba et al., 2019) with a TPE sampler (Bergstra et al., 2011) (4 random start-up trials; a constant-liar strategy lets concurrent trials of one study run in parallel) and a median pruner. Each study runs 12 trials at $N = 3$. A trial trains the sampled configuration on 3 tuning seeds $\{7, 77, 777\}$ $\times$ 3 tuning variations $\{1, 4, 7\}$ (nominal winds of 1.0, 1.5 and 3.0 m/s from three different directions), with the full 2×10^6 step budget and exactly the training code of the final runs; each of the nine runs is scored on 50 held-out episodes, and the trial's objective is the *median* of the nine held-out IQMs, so a configuration must work across seeds and field layouts rather than on one lucky run. The tuning seeds are disjoint from the evaluation seeds. At 25%, 50% and 75% of the budget a trial reports the median over its nine runs of the mean of their last three checkpoint-selection evaluations, and after 4 completed trials the pruner stops a trial whose value falls below the median of earlier trials at the same budget. The winning configuration is fixed for all tuned runs at every team size: tuning ran at $N = 3$ only, so the tuned cells of Table 1 at $N \in \{2, 4, 5\}$ and the scalability study transfer these hyperparameters to the other team sizes without re-tuning. If the winning trial's objective is not positive, no trial learned the task, and the tuned cell is marked * as the best of a set of failures rather than a tuned result. The off-policy `gradient_steps` of VDN is searched over $\{1, 2\}$ because run cost grows almost linearly with it.

The search spaces are listed in Table 9; the selected configurations are in Tables 10 and 11, and those of the model-based controller in Table 8. PPO, TRPO and MAPPO all select learning rates near the lower end of their ranges (1–2×10^{-4}). TRPO also selects the tightest trust region it can find (`target_kl` $= 1.8 \times 10^{-3}$) and the maximum number of conjugate-gradient steps. Even so, its winning trial fails on 4 of its 9 runs, 3 of them on variation 7, and its objective (2868) is the lowest of all studies. Tuned TRPO is therefore still unreliable (Table 1). MAPPO selects few epochs (3) with 4 minibatches, i.e. conservative policy updates. VDN selects the finest action grid (7 thrust bins, 147 actions), the slowest target update (every 1000 steps), a short exploration schedule (5.6×10^4 steps) with a high final ϵ (0.17), and small batches. Its objective (3565) is close to MAPPO's, yet the tuned configuration scores below the default one on the evaluation variations (Table 1): it was

Table 9: Optuna search spaces. $\mathcal{LU}$ denotes a log-uniform continuous range, $\mathcal{U}$ a uniform one, and braces a categorical or integer set. Parameters not listed for an algorithm are held at their default.

Algorithm	Hyperparameter	Range
PPO, TRPO	`learning_rate`	$\mathcal{LU}(10^{-4},\ 10^{-2})$
PPO, TRPO	`gae_lambda`	$\mathcal{U}(0.90,\ 1.00)$
PPO	`vf_coef`	$\mathcal{U}(0.20,\ 0.70)$
PPO	`ent_coef`	$\mathcal{U}(0.00,\ 0.05)$
PPO	`max_grad_norm`	$\mathcal{U}(0.30,\ 0.99)$
PPO	`clip_range`	$\mathcal{U}(0.10,\ 0.40)$
PPO	`n_epochs`	$\{3, \dots, 20\}$
TRPO	`target_kl`	$\mathcal{U}(10^{-3},\ 5 \times 10^{-2})$
TRPO	`cg_max_steps`	$\{5, \dots, 20\}$
MAPPO, VDN	`lr`	$\mathcal{LU}(10^{-5},\ 10^{-2})$
MAPPO	`critic_lr`	$\mathcal{LU}(10^{-5},\ 10^{-2})$
MAPPO	`clip`	$\mathcal{U}(0.10,\ 0.40)$
MAPPO	`gae_lambda`	$\mathcal{U}(0.90,\ 1.00)$
MAPPO	`entropy_coef`	$\mathcal{U}(0.00,\ 0.05)$
MAPPO	`ppo_epochs`	$\{3, \dots, 20\}$
MAPPO	`num_minibatch`	$\{1, 2, 4\}$
MAPPO	`rollout`	$\{100, 200, 400\}$
VDN	`batch_size`	$\{128, 256, 512\}$
VDN	`buffer_size`	$\mathcal{LU}(5 \times 10^4,\ 5 \times 10^5)$
VDN	`gradient_steps`	$\{1, 2\}$
VDN	`target_update_interval`	$\{100, 200, 500, 1000\}$
VDN	`eps_end`	$\mathcal{U}(0.01,\ 0.20)$
VDN	`eps_decay_steps`	$\mathcal{LU}(5 \times 10^4,\ 5 \times 10^5)$
VDN	`thrust_bins`	$\{3, 5, 7\}$ (27, 75, 147 actions)

Table 10: Best hyperparameters found by Optuna for the centralized baselines (12 trials; $N = 3$, 3 seeds $\times$ 3 variations per trial, 2×10^6 steps per run). N/A = not searched for the algorithm. The last rows give the tuning objective of the winning trial (median over its nine runs of the held-out IQM) and the number of complete / pruned trials.

Hyperparameter	PPO	TRPO
Learning rate	1.98×10^{-4}	1.02×10^{-4}
GAE λ	0.909	0.999
`vf_coef`	0.690	N/A
`ent_coef`	0.0165	N/A
`max_grad_norm`	0.594	N/A
`clip_range`	0.350	N/A
`n_epochs`	14	N/A
`target_kl`	N/A	1.81×10^{-3}
`cg_max_steps`	N/A	20
Tuning objective (median IQM)	3562	2868
Trials complete / pruned	7 / 5	9 / 3

selected on three variations and does not carry over to the other seven. The PPO, MAPPO and VDN studies pruned 5, 2 and 6 of their 12 trials, respectively. Tuning makes the model-based controller faster (`v_cmd` 2.0 $\rightarrow$ 4.83 m/s) and more spread out (larger repulsion, clustering and spraying radii), which raises its objective from 3148 to 3338.

E.2 Extended Main Results

Table 13 gives Table 1 with the 95% stratified-bootstrap interval of every cell and adds TQC with default hyperparameters. TQC completes the task only at $N = 2$ (2270) and degrades steeply with the team size (-11254 at $N = 5$), like the other centralized learners. Section E.3 reports an extended benchmark of ten learners, in which tuned TQC is the best centralized learner at $N = 3$. Table 14

Table 11: Best hyperparameters found by Optuna for the MARL baselines (protocol as in Table 10). "–" = not searched for that algorithm.

Hyperparameter	MAPPO	VDN
`lr`	1.22×10^{-4}	1.31×10^{-3}
`critic_lr`	8.69×10^{-4}	–
`clip`	0.293	–
`gae_lambda`	0.936	–
`entropy_coef`	0.0162	–
`ppo_epochs`	3	–
`num_minibatch`	4	–
`rollout`	200	–
`batch_size`	–	128
`buffer_size`	–	52,148
`gradient_steps`	–	1
`target_update_interval`	–	1000
`eps_end`	–	0.167
`eps_decay_steps`	–	55,737
`thrust_bins`	–	7
Tuning objective (median IQM)	3582	3565
Trials complete / pruned	10 / 2	6 / 6

Table 12: Default hyperparameters for the MARL baselines — the multi-agent counterpart of the Stable-Baselines3 defaults used for the centralized block. Both use 8 parallel environments, $\gamma = 0.99$, two hidden layers of 64 units, gradient-norm clipping at 10.0, and a per-agent one-hot index appended to the actor and critic inputs. "–" = not used by the algorithm.

Hyperparameter	MAPPO	VDN
Actor / utility learning rate	5×10^{-4}	5×10^{-4}
Critic learning rate	5×10^{-4}	–
Rollout length	200	–
GAE λ	0.95	–
Clip range	0.2	–
Epochs / minibatches	10 / 1	–
Entropy coefficient	0.01	–
Value coefficient	1.0	–
$\log \sigma$ init	-0.5	–
Value normalization	yes	–
LR annealing	linear	–
Buffer size	–	10^5
Batch size	–	256
Learning starts	–	5000
Gradient steps per iteration	–	1
Target update interval	–	200 (hard)
ϵ-greedy schedule	–	$1.0 \to 0.05$ / 3×10^5
Double Q	–	yes
Reward scale (buffer only)	–	0.01
Action grid	continuous	$5 \times 5 \times 3$

compares the selection-phase and held-out IQMs of the same checkpoints. Fig. 3 shows the learning curves with default hyperparameters.

Section 6 discusses the main results; the intervals add the following. They are tight wherever every run of a cell converges, e.g. $[3527, 3537]$ for tuned MAPPO at $N = 3$. They are wide wherever a fraction of the runs fails: $[2710, 3505]$ for default PPO at $N = 4$, $[1361, 2996]$ for default TQC at $N = 2$, and $[1238, 2935]$ and $[1718, 2884]$ for MAPPO at $N = 2$. The advantage of default over tuned VDN is outside the intervals at $N = 5$ ($[3250, 3481]$ against $[2304, 2834]$) but not at $N \leq 4$. Best-checkpoint selection is optimistic (Table 14). The gap between the selection-phase and held-out IQMs is below 40 for tuned PPO and MAPPO at $N \geq 3$, but it grows where runs fail, up to 566

Table 13: Table 1 in full: held-out IQM over 5 seeds × 10 variations with the 95% stratified-bootstrap interval. Hyperparameters are tuned at $N = 3$ and transferred to the other team sizes. The model-based controller is listed with its untuned parameters under the default columns and with its tuned parameters under the tuned columns. TQC is reported in this appendix only, with default hyperparameters ("—": not run). **Bold**: best per column, as in Table 1; †: IQM above 1700.

Method	Default hyperparameters				Tuned hyperparameters			
	$N = 2$	$N = 3$	$N = 4$	$N = 5$	$N = 2$	$N = 3$	$N = 4$	$N = 5$
PPO	**3491**† [3484, 3498]	**3514**† [3503, 3523]	3287† [2710, 3505]	−255 [−739, 214]	**3484**† [3470, 3496]	**3534**† [3528, 3539]	3549† [3542, 3555]	**3527**† [3477, 3540]
TRPO	−21059 [−26911, −15137]	−25265 [−31148, −20014]	−16220 [−22342, −11204]	−1438 [−1531, −1032]	3365† [2917, 3459]	1755† [681, 2556]	−858 [−3253, 209]	−1516 [−1567, −1433]
TQC	2270† [1361, 2996]	744 [−38, 1539]	−1472 [−2815, −630]	−11254 [−17601, −6254]	—	—	—	—
MAPPO	2127† [1238, 2935]	3510† [3479, 3522]	**3543**† [3514, 3555]	**3549**† [3535, 3559]	2310† [1718, 2884]	3533† [3527, 3537]	**3551**† [3539, 3560]	3519† [3506, 3529]
VDN	2927† [2244, 3293]	3451† [3167, 3494]	3455† [3336, 3512]	3416† [3250, 3481]	2369† [1715, 2875]	3144† [2622, 3373]	3335† [3070, 3403]	2626† [2304, 2834]
Model-based	2042† [1943, 2136]	2789† [2728, 2837]	2479† [2371, 2563]	2072† [1923, 2197]	2651† [2544, 2715]	2926† [2875, 2973]	2894† [2826, 2948]	1984† [1840, 2109]

Table 14: Optimism of best-checkpoint selection: selection-phase IQM (best of the 5-episode checkpoint evaluations) / held-out IQM (the same checkpoint on 50 fresh episodes), over 5 seeds × 10 variations. "—": not run.

Algorithm	Default hyperparameters				Tuned hyperparameters			
	$N = 2$	$N = 3$	$N = 4$	$N = 5$	$N = 2$	$N = 3$	$N = 4$	$N = 5$
PPO	3496 / 3491	3518 / 3514	3394 / 3287	−152 / −255	3499 / 3484	3539 / 3534	3555 / 3549	3547 / 3527
TRPO	−1665 / −21059	−1668 / −25265	−1473 / −16220	−1362 / −1438	3390 / 3365	1966 / 1755	−391 / −858	−1442 / −1516
TQC	2537 / 2270	1275 / 744	−953 / −1472	−1434 / −11254	—	—	—	—
MAPPO	2367 / 2127	3519 / 3510	3553 / 3543	3562 / 3549	2427 / 2310	3538 / 3533	3565 / 3551	3556 / 3519
VDN	3237 / 2927	3484 / 3451	3506 / 3455	3491 / 3416	2856 / 2369	3391 / 3144	3413 / 3335	3192 / 2626

among the tuned runs (VDN at $N = 5$). With default hyperparameters, TRPO's held-out IQM is an order of magnitude below its selection-phase IQM (e.g. −25265 against −1668 at $N = 3$). The selected best of many 5-episode evaluations is far more favorable than the 50 held-out episodes of the same checkpoint, so selection on a few episodes can rank a failing run far too high. The default-hyperparameter learning curves (Fig. 3) show the same picture over training. MAPPO crosses the threshold within the first 0.3–0.6 × 10^6 steps at every $N \geq 3$ and stays there, and PPO converges at $N \leq 4$ but only slowly at $N = 4$ and not at $N = 5$. The per-step IQMs of TRPO and TQC stay below the threshold throughout. The checkpoint-selection return of VDN is far below its held-out IQM for most of training and fluctuates strongly. VDN's held-out scores therefore come from the best checkpoint, not from a stable policy, which also explains its large selection gaps in Table 14.

E.3 Extended Benchmark: Ten Learners

To test whether the conclusions of the main paper depend on the choice of baselines, we ran a broader benchmark as a separate, independent set of runs. It uses the same environment, calibrated kernel, training code, evaluation protocol, seeds and variations as the main experiments. On the centralized POMDP, it adds A2C (Mnih et al., 2016), ARS (Mania et al., 2018), TQC (Kuznetsov et al., 2020) and CrossQ (Bhatt et al., 2024) to PPO and TRPO. On the Dec-POMDP, it adds MADDPG (Lowe

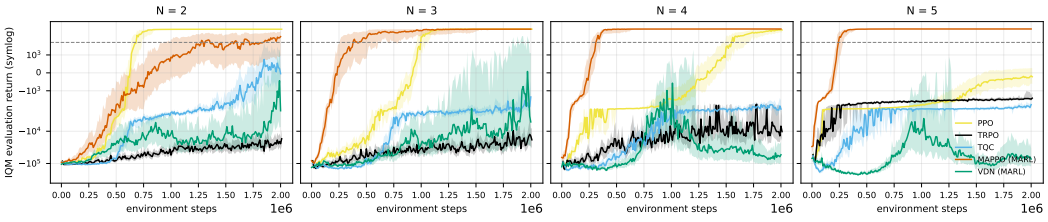


Figure 3: Learning curves with default hyperparameters: IQM over 5 seeds × 10 variations of the checkpoint-selection return as a function of environment steps, with 95% stratified-bootstrap bands; one panel per team size. The vertical axis is symmetric-logarithmic. The dashed line is the success threshold of 1700.

et al., 2017) and QMIX (Rashid et al., 2018) to MAPPO and VDN. Every learner was tuned in its own 12-trial study at $N = 3$ (independent of the studies of Section E.1), and the tuned configurations were evaluated on the main grid at $N = 3$ only (the scalability study below transfers them to other team sizes). The ablations, the observation gap and the scalability study of this benchmark compare the tuned centralized TQC with the tuned decentralized MAPPO. The model-based controller of this earlier run acted on the true state rather than on the observation, so it is not comparable with the controller of Section D and is omitted here. Because PPO, TRPO, MAPPO and VDN appear in both sets of runs, this benchmark also serves as a replication.

Main results (Tables 15 and 16). With default hyperparameters, the results of the four learners shared with Table 1 have overlapping confidence intervals at every team size. For example, PPO scores 3518 against 3514 at $N = 3$ and collapses at $N = 5$ in both sets of runs (325 and -255), and MAPPO scores 3547–3550 against 3543–3549 at $N \in \{4, 5\}$. CrossQ is the strongest centralized learner with default hyperparameters: it is the only one above the threshold at every team size (3387–3490) and the best at $N \in \{4, 5\}$. TQC completes the task only at $N = 2$ (1810), as in Table 13. A2C, ARS, TRPO and QMIX fail at every team size, and MADDPG clears the threshold only at $N = 5$ (1745, with an interval reaching -700). Tuning succeeds for PPO, TQC, CrossQ, MAPPO and VDN, whose tuned $N = 3$ IQMs all lie between 3499 and 3521, with tuned TQC the highest. No trial of A2C, ARS, TRPO or QMIX learned the task (Table 16). TRPO's study here found no working configuration, whereas the study of Table 10 did, which shows how sensitive TRPO is to its hyperparameters. MADDPG's winning trial reached 2792 on the tuning cells, yet it scores -338 on the evaluation cells, a failure to transfer similar to that of tuned VDN in Table 1. The tuned configurations of the shared learners also differ from those of the main runs, and so do their scores (tuned VDN scores 3518 here against 3144 there), which shows that a single 12-trial study can land on configurations of quite different quality.

Ablations (Tables 17 and 18). The ablations reproduce the main findings for MAPPO and extend them to a centralized off-policy learner. Removing the terminal signal again raises MAPPO's collision rate, here to 43%, while TQC keeps completing essentially all episodes ($\geq 99.7\%$) under every reward ablation, with almost unchanged episode lengths, so it does not depend on any single reward term. Removing the path cost lengthens MAPPO's episodes ($17.8 \to 44.0$ steps) but not TQC's. Among the observation blocks, own position is again indispensable for both learners (TQC -19918, MAPPO -1334), the infection map is again redundant, and position alone suffices for MAPPO (3391) and nearly for TQC (3225). The noise sources and the wind model change neither learner's IQM by more than 1% (TQC 3430–3466, MAPPO 3393–3415). The TQC run under wind-only noise did not complete and is not reported.

Observation gap (Tables 19 and 20). The centralized TQC is far more fragile to inference-time sensor perturbations than the decentralized MAPPO. Position noise of 5 m lowers TQC's IQM from 3521 to 2517 (median per-run drop 23%), against $3505 \to 3378$ for MAPPO (1.3%). Under the combined perturbation, TQC collapses to -6130 (median drop 250%), while MAPPO still scores 2949 (9.3%). TQC's drop is significantly larger under every perturbation ($p \leq 0.005$). Within TQC, position noise hurts more than staleness ($p = 3.8 \times 10^{-4}$); within MAPPO, the two do not differ ($p = 0.18$). This is the same ordering as in Table 3: the decentralized actor–critic learner is the most robust.

Scalability (Table 21). TQC is above the threshold up to $N = 6$ (3340), but its centralized network, which grows with the team, degrades at $N = 7$ (1969, with an interval reaching -1365) and fails at $N = 8$ (739). MAPPO with $\nu = 4$ fails at $N = 2$ (965), as it is weakest at $N = 2$ in Table 1, but it holds 3470 and 3419 at $N = 7, 8$, and a full neighbor block ($\nu = N - 1$) does not improve on this. MAPPO's tuned configuration here scales better than the one in Table 28 (1843 and 1800 at $N = 7, 8$), so its behavior at large team sizes depends on the hyperparameters found at $N = 3$.

E.4 Extended Ablation Study

All ablations are run at $N = 3$ on the two decentralized learners, MAPPO and VDN, with their tuned hyperparameters, and on the tuned model-based controller. The completion threshold is relaxed to 90% of the initial infection ($\ell_\epsilon = 0.1 \sum_j \ell_j(0)$) throughout this subsection, so that conditions which

Table 15: Extended benchmark of ten learners (an independent set of runs; see the text): held-out IQM over 5 seeds × 10 variations with the 95% stratified-bootstrap interval. Tuned hyperparameters were found and evaluated at $N = 3$ only. **Bold**: best per column within each block; †: IQM above 1700; *: no tuning trial learned the task (best of failures).

Algorithm	Default hyperparameters				Tuned
	$N = 2$	$N = 3$	$N = 4$	$N = 5$	$N = 3$
Single-agent (centralized POMDP)					
A2C	-25587 [$-32806, -17529$]	-29360 [$-35635, -22431$]	-30622 [$-38659, -24039$]	-16682 [$-23178, -11099$]	-14276 * [$-24919, -6963$]
ARS	-16460 [$-18959, -14291$]	-8805 [$-11245, -7211$]	-3454 [$-5536, -1959$]	-1053 [$-2995, -40$]	-11841 * [$-14558, -9445$]
PPO	$\mathbf{3484}^{\dagger}$ [3466, 3496]	$\mathbf{3518}^{\dagger}$ [3507, 3527]	$3235^{\dagger}$ [2481, 3466]	325 [-8, 821]	$3508^{\dagger}$ [3501, 3515]
TRPO	-19788 [$-24385, -14305$]	-19725 [$-26547, -14216$]	-11706 [$-18573, -7893$]	-1549 [$-2199, -1361$]	-652 * [-2772, 816]
TQC	$1810^{\dagger}$ [914, 2668]	-58 [-851, 815]	-4094 [$-9111, -1627$]	-13301 [$-21320, -7040$]	$\mathbf{3521}^{\dagger}$ [3506, 3533]
CrossQ	$3446^{\dagger}$ [3425, 3458]	$3470^{\dagger}$ [3446, 3488]	$\mathbf{3490}^{\dagger}$ [3463, 3508]	$\mathbf{3387}^{\dagger}$ [3085, 3448]	$3499^{\dagger}$ [3486, 3511]
Multi-agent (Dec-POMDP)					
MAPPO	$2257^{\dagger}$ [1262, 3096]	$\mathbf{3493}^{\dagger}$ [3350, 3514]	$\mathbf{3547}^{\dagger}$ [3534, 3557]	$\mathbf{3550}^{\dagger}$ [3540, 3558]	$3509^{\dagger}$ [3482, 3525]
MADDPG	1388 [-5191, 2405]	-1534 [$-3386, -88$]	-272 [-1943, 714]	$1745^{\dagger}$ [-700, 2544]	-338 [-1269, 584]
VDN	$\mathbf{3222}^{\dagger}$ [2748, 3407]	$3407^{\dagger}$ [3227, 3468]	$3505^{\dagger}$ [3421, 3534]	$3395^{\dagger}$ [3284, 3439]	$\mathbf{3518}^{\dagger}$ [3506, 3526]
QMIX	-52116 [$-60013, -44359$]	-56342 [$-66451, -47266$]	-44778 [$-57429, -34954$]	-36536 [$-48364, -26595$]	-32206 * [$-43780, -23020$]

Table 16: Tuning studies of the extended benchmark (12 trials each at $N = 3$, 3 seeds × 3 variations per trial): objective of the winning trial (median over its nine runs of the held-out IQM), whether any trial learned the task (objective > 0), and complete / pruned trials.

Algorithm	Objective	Learned	Complete / pruned
A2C	-1443	no	8 / 4
ARS	-1907	no	11 / 1
PPO	3551	yes	5 / 7
TRPO	-1410	no	5 / 7
TQC	3579	yes	11 / 1
CrossQ	3545	yes	9 / 3
MAPPO	3572	yes	6 / 6
MADDPG	2792	yes	10 / 2
VDN	3571	yes	7 / 5
QMIX	-1964	no	8 / 4

learn a partial policy still terminate and their episode lengths remain interpretable. Unless a row says otherwise, training and evaluation both use the full stochastic noise model and the correlated wind. The unablated configuration is trained once, with 5 seeds × 10 variations, and this single set of runs provides the "Full" row of every block, so those rows are identical and directly comparable. The reward and observation ablations, whose conditions change the task drastically, use 3 seeds $\{0, 42, 123\}$ × 10 variations per condition; the uncertainty and wind ablations use 5 seeds × 10 variations. Every cell is scored on the held-out episodes of Section E.1.

The model-based controller is not trained, so it is *evaluated* under each condition on the same cells, held-out episodes and completion threshold as the RL runs. Its behavior does not depend on the reward, so its reward rows differ from the Full row only through the seed set; its two wind rows,

Table 17: Ablations of the extended benchmark at $N = 3$ (90% completion threshold) for tuned TQC and tuned MAPPO; protocol as in Section E.4. Reward block: episode-length IQM and sprayed / collision (%). Observation and uncertainty blocks: held-out IQM with 95% interval. "—": not reported (the TQC run under wind-only noise did not complete). †: IQM above 1530.

Reward ablation					Observation ablation			Uncertainty ablation		
Condition	TQC		MAPPO		Condition	TQC	MAPPO	Condition	TQC	MAPPO
	Ep. len.	Spr. / coll.	Ep. len.	Spr. / coll.						
Full reward	17.7	100 / 0	17.8	100 / 0	Full	$3433^{\dagger}$ [3426, 3439]	$3409^{\dagger}$ [3398, 3417]	Full	$3433^{\dagger}$ [3426, 3439]	$3409^{\dagger}$ [3398, 3417]
No r_{term}	19.0	100 / 0	17.0	57 / 43	No pos.	-19918 [−25775, −15693]	-1334 [−11393, 1325]	Wind only	—	$3415^{\dagger}$ [3399, 3424]
No r_{spray}	16.9	100 / 0	17.4	97 / 3	No infect.	$3395^{\dagger}$ [3380, 3405]	$3413^{\dagger}$ [3399, 3423]	Act. only	$3449^{\dagger}$ [3436, 3462]	$3393^{\dagger}$ [3377, 3406]
No r_{path}	18.5	100 / 0	44.0	86 / 0	Pos. only	$3225^{\dagger}$ [3161, 3280]	$3391^{\dagger}$ [3377, 3404]	Obs. only	$3436^{\dagger}$ [3426, 3445]	$3410^{\dagger}$ [3391, 3425]
								Deterministic	$3466^{\dagger}$ [3453, 3476]	$3412^{\dagger}$ [3394, 3426]

Table 18: Wind-model ablation of the extended benchmark ($N = 3$, tuned hyperparameters, 90% completion threshold): held-out IQM of policies trained under the row wind model and evaluated under the column wind model. †: IQM above 1530.

Trained under	TQC		MAPPO	
	eval AR(1)	eval i.i.d.	eval AR(1)	eval i.i.d.
AR(1)	$3433^{\dagger}$ [3426, 3439]	$3430^{\dagger}$ [3421, 3437]	$3409^{\dagger}$ [3398, 3417]	$3409^{\dagger}$ [3398, 3418]
i.i.d.	$3432^{\dagger}$ [3420, 3445]	$3438^{\dagger}$ [3428, 3449]	$3412^{\dagger}$ [3399, 3422]	$3413^{\dagger}$ [3400, 3423]

and every entry of its uncertainty cross-evaluation, are single evaluations; and its observation rows exercise the observer of Section D.

Reward function (Table 22). Removing a reward term changes the scale of the return, so within this block we report the IQM of held-out episode *length* together with the fractions of episodes that end with the field sprayed and in a collision. A short episode can be a fast solution or an early collision; the terminal columns disambiguate the two. The Full rows show that the tuned learners finish in 17.6 (MAPPO) and 22.7 (VDN) steps, against the 54.3 steps of the model-based controller. Without the terminal signal, MAPPO keeps its speed but 17% of its episodes end in a collision, whereas VDN keeps its completion rate and slows down. Without the spray reward, both learners still finish, from the terminal signal alone, but less reliably (93% and 88%), and VDN takes almost twice as long. Without the path cost, there is no pressure to finish quickly: MAPPO's episodes lengthen from 17.6 to 63.3 steps and VDN's to 395.9, and VDN completes only 56% of them. The

Table 19: Observation gap of the extended benchmark ($N = 3$, tuned hyperparameters, 5 seeds × 10 variations, 100 held-out episodes per run and condition; protocol of Section E.6): held-out IQM with 95% interval, mean per-run relative change Δ and percentage of episodes that end sprayed. †: IQM above 1700.

Perturbation	TQC (centralized)			MAPPO (decentralized)		
	IQM	Δ (%)	Sprayed (%)	IQM	Δ (%)	Sprayed (%)
None (baseline)	$3521^{\dagger}$ [3506, 3532]	—	99.5	$3505^{\dagger}$ [3481, 3522]	—	91.5
Position noise ($\sigma = 5.0$ m)	$2517^{\dagger}$ [1752, 2894]	−129.4	89.6	$3378^{\dagger}$ [3256, 3432]	−0.9	89.3
Velocity staleness (3 steps)	$3314^{\dagger}$ [3022, 3399]	−56.6	92.7	$3436^{\dagger}$ [3341, 3477]	−4.5	89.4
Field-map loss (10%)	$3387^{\dagger}$ [3250, 3452]	−18.3	97.1	$3498^{\dagger}$ [3443, 3520]	−17.7	89.3
All combined (staleness 5)	-6130 [−9678, −2811]	−441.6	69.8	$2949^{\dagger}$ [2619, 3126]	−44.8	81.7

Table 20: Paired two-sided Wilcoxon signed-rank tests on the per-run relative drop of the extended benchmark's observation-gap study (pairs = (variation, seed) runs; medians of the per-run relative change in %).

Test	**Condition**	**Pairs**	**Medians**	W	p
TQC vs. MAPPO	Position noise	50	-23.01 / -1.28	116	3.3×10^{-8}
TQC vs. MAPPO	Velocity staleness	50	-4.29 / -0.77	350	0.005
TQC vs. MAPPO	Field-map loss	50	-2.55 / -0.03	242	7.2×10^{-5}
TQC vs. MAPPO	All combined	50	-250.40 / -9.32	63	2.5×10^{-10}
TQC: position vs. staleness	—	50	-23.01 / -4.29	280	3.8×10^{-4}
MAPPO: position vs. staleness	—	50	-1.28 / -0.77	498	0.18

Table 21: Scalability of the extended benchmark on variations 1, 5 and 9 (5 seeds each, 50 held-out episodes per run, hyperparameters tuned at $N = 3$): held-out IQM with 95% interval. "—": not run. †: IQM above 1700.

Method	$N = 2$	$N = 3$	$N = 4$	$N = 5$	$N = 6$	$N = 7$	$N = 8$
TQC	$3488^{\dagger}$ [3479, 3496]	$3530^{\dagger}$ [3511, 3539]	$3550^{\dagger}$ [3541, 3558]	$3535^{\dagger}$ [3494, 3548]	$3340^{\dagger}$ [2791, 3493]	$1969^{\dagger}$ [−1365, 2835]	739 [−1479, 1704]
MAPPO ($\nu = 4$)	965 [−102, 2092]	$3504^{\dagger}$ [3472, 3520]	$3526^{\dagger}$ [3502, 3546]	$3516^{\dagger}$ [3500, 3536]	$3534^{\dagger}$ [3503, 3547]	$3470^{\dagger}$ [3354, 3510]	$3419^{\dagger}$ [3280, 3477]
MAPPO ($\nu = N - 1$)	—	—	—	—	$3532^{\dagger}$ [3522, 3540]	$3432^{\dagger}$ [3378, 3478]	$3346^{\dagger}$ [3245, 3428]

model-based controller's three reduced-reward rows are identical, since its behavior does not depend on the reward, and differ from its Full row only because they cover 3 seeds rather than 5.

Observation space (Table 23). Each condition drops whole blocks of the observation (Table 7); Δ is the relative change of the IQM against the Full row. Each robot's own position is the one indispensable block. Without it, MAPPO's IQM falls to -28935 (65% of episodes end in a collision, 16% are completed) and VDN's to -1778. The infection map is nearly redundant for both learners (-0.2% and -0.1%). The sites of a variation are fixed and every run is trained on one variation, so a policy can learn where the sites are instead of reading them from the map. Position alone (Pos. only) suffices for MAPPO (-0.8%) but not for VDN (-93%), which needs at least one of the other blocks. The model-based controller loses nothing under any condition, because its observer replaces the dropped blocks. It even gains 8–9% when the map is removed: the observer then predicts the remaining infection from its own spray commands with the noise-free kernel, and this prediction is evidently more useful to it than the noisy map smoothed exponentially.

Physical uncertainty (Tables 24 and 26). Each policy is trained and evaluated under the matching noise condition of Table 25; the observation-noise-only condition isolates the sensor noise, which the full condition otherwise confounds with the process noise. In addition, the policies trained under each condition are cross-evaluated under all five conditions on the same held-out episodes (Table 26), which separates the effect of training under a noise source from the effect of being tested under it. At the calibrated magnitudes, no noise source matters much to the learners. MAPPO's five diagonal cells lie within 3416–3429, and VDN's within 3303–3370 with overlapping intervals. The cross-evaluation is equally flat: every MAPPO policy scores 3411–3430 and every VDN policy 3302–3380 under every test condition. Policies trained without noise therefore lose nothing when tested with it, and no policy is specialized to its training condition. The model-based controller is the most sensitive method. It scores 3102 without noise and 2921 with full noise. With a single source active, it scores 3073 (wind), 2990 (actuation) and 2960 (observation), so the actuation and observation noise cost it more than the wind does.

Wind model (Table 27). Policies are trained under the correlated wind (the Full runs) or under independently resampled wind ($\phi = 0$) with the same stationary marginal, and each is evaluated under both, so the table separates the effect of temporal correlation at training time from its effect at test time. The wind model makes no measurable difference. MAPPO scores 3416–3418 in all four train/test combinations and VDN 3353–3371, with overlapping intervals, and the model-based

Table 22: Reward-function ablation at $N = 3$, 90% completion threshold. Full: 5 seeds × 10 variations; other conditions: 3 seeds × 10 variations; 50 held-out episodes per run. Primary metric: IQM of episode length, read together with the percentage of episodes that end sprayed / in a collision — a short episode that ends in a collision is a failure, not a fast solution.

Condition	Term removed	MAPPO		VDN		Model-based	
		Ep. len. IQM	Sprayed / coll. (%)	Ep. len. IQM	Sprayed / coll. (%)	Ep. len. IQM	Sprayed / coll. (%)
Full reward	—	17.6	100 / 0	22.7	95 / 4	54.3	96 / 4
No r_{term}	$\pm r_L$ terminal signal	18.1	83 / 17	28.8	96 / 4	54.6	96 / 4
No r_{spray}	spray reward	17.2	93 / 5	41.1	88 / 10	54.6	96 / 4
No r_{path}	stage cost $\mathcal{J}$ + out-of-field	63.3	83 / 1	395.9	56 / 11	54.6	96 / 4

Table 23: Observation-space ablation at $N = 3$, 90% completion threshold: held-out IQM with 95% stratified-bootstrap interval. Full: 5 seeds × 10 variations; other conditions: 3 seeds × 10 variations. Δ: relative change with respect to the full observation. Dimensions are given in Table 7. †: IQM above 1530.

Condition	MAPPO		VDN		Model-based	
	IQM	Δ (%)	IQM	Δ (%)	IQM	Δ (%)
Full	$3416^{\dagger}$ $[3412, 3420]$	—	$3353^{\dagger}$ $[3319, 3377]$	—	$2921^{\dagger}$ $[2879, 2960]$	—
No pos.	-28935 $[-37693, -20605]$	-947.1	-1778 $[-3291, -870]$	-153.0	$2951^{\dagger}$ $[2849, 2988]$	$+1.0$
No infect.	$3410^{\dagger}$ $[3402, 3414]$	-0.2	$3349^{\dagger}$ $[3287, 3363]$	-0.1	$3171^{\dagger}$ $[3121, 3189]$	$+8.6$
Pos. only	$3388^{\dagger}$ $[3376, 3399]$	-0.8	234 $[-1692, 957]$	-93.0	$3163^{\dagger}$ $[3145, 3180]$	$+8.3$

controller scores 2921 under AR(1) wind and 2953 under i.i.d. wind. At $\tau_w = 20$ steps and the calibrated σ_w, the correlated wind does not make the task harder, and a policy trained under it does not rely on the correlation.

E.5 Scalability

The scalability study trains MAPPO and VDN at $N \in \{2, \dots, 8\}$ on variations 1, 5 and 9 (the larger fields) with 5 seeds each and the full evaluation protocol. Tuned hyperparameters exist only at $N = 3$, so every other team size uses the $N = 3$-tuned hyperparameters *transferred* without re-tuning; the $N \leq 5$ cells are the tuned Table 1 runs on the same three variations. Both learners are run with the default neighbor budget $\nu = 4$ and, for $N \in \{6, 7, 8\}$, with $\nu = N - 1$ (Section C.6); for $N \leq 5$ the two coincide in information. The tuned model-based controller is evaluated on the same cells. The centralized learners are not included beyond Table 1 ($N \leq 5$): their observation and action dimensions grow with N, so their networks grow with the team. Table 28 and Fig. 4 give the results.

Table 24: Physical-uncertainty ablation at $N = 3$, 5 seeds × 10 variations, 50 held-out episodes per run, 90% completion threshold. Each policy is trained and evaluated under the matching noise condition (Table 25); the model-based controller is evaluated under it. Held-out IQM with 95% stratified-bootstrap interval. †: IQM above 1530.

Train / eval condition	MAPPO	VDN	Model-based
Full noise	$3416^{\dagger}$ $[3412, 3420]$	$3353^{\dagger}$ $[3319, 3377]$	$2921^{\dagger}$ $[2879, 2960]$
Wind only	$3426^{\dagger}$ $[3418, 3430]$	$3346^{\dagger}$ $[3262, 3373]$	$3073^{\dagger}$ $[3033, 3096]$
Actuation only	$3418^{\dagger}$ $[3411, 3421]$	$3303^{\dagger}$ $[3170, 3342]$	$2990^{\dagger}$ $[2936, 3022]$
Observation only	$3429^{\dagger}$ $[3426, 3433]$	$3370^{\dagger}$ $[3269, 3385]$	$2960^{\dagger}$ $[2913, 2999]$
Deterministic	$3423^{\dagger}$ $[3411, 3431]$	$3338^{\dagger}$ $[3054, 3368]$	$3102^{\dagger}$ $[3059, 3121]$

Table 25: Noise standard deviations activated by each uncertainty mode. Symbols and their roles are defined in Table 4. "Actuation" covers both the force noise ϵ_a of Eq. 3 and the position process noise ϵ_p of Eq. 4; "Observation" covers the sensor noise of Eq. 16 and its local counterpart Eq. 21. With $\sigma_w = \sigma_\theta = 0$ the wind is constant at its nominal value.

Mode	σ_w	σ_θ	σ_a	σ_p	σ_s	σ_{obs}	$\sigma_{w,obs}$
Full	0.20	5.0°	0.10	0.05	2×10^{-3}	0.01	0.02
Wind only	0.20	5.0°	0	0	0	0	0
Actuation only	0	0	0.10	0.05	0	0	0
Observation only	0	0	0	0	0	0.01	0.02
Deterministic	0	0	0	0	0	0	0

Table 26: Cross-evaluation of the uncertainty ablation ($N = 3$, 5 seeds $\times$ 10 variations, 50 held-out episodes per run and condition, 90% completion threshold): held-out IQM of policies trained under the row condition and evaluated under the column condition. The diagonal reproduces Table 24. The model-based controller is not trained and has a single row.

	Trained under	**Evaluated under** Full	Wind only	Act. only	Obs. only	Determ.
MAPPO	Full	$3416^\dagger$	$3415^\dagger$	$3413^\dagger$	$3419^\dagger$	$3419^\dagger$
	Wind only	$3420^\dagger$	$3426^\dagger$	$3411^\dagger$	$3429^\dagger$	$3430^\dagger$
	Act. only	$3422^\dagger$	$3420^\dagger$	$3418^\dagger$	$3421^\dagger$	$3421^\dagger$
	Obs. only	$3423^\dagger$	$3425^\dagger$	$3419^\dagger$	$3429^\dagger$	$3429^\dagger$
	Deterministic	$3417^\dagger$	$3415^\dagger$	$3411^\dagger$	$3423^\dagger$	$3423^\dagger$
VDN	Full	$3353^\dagger$	$3340^\dagger$	$3338^\dagger$	$3360^\dagger$	$3362^\dagger$
	Wind only	$3347^\dagger$	$3346^\dagger$	$3352^\dagger$	$3344^\dagger$	$3349^\dagger$
	Act. only	$3319^\dagger$	$3309^\dagger$	$3303^\dagger$	$3325^\dagger$	$3330^\dagger$
	Obs. only	$3372^\dagger$	$3370^\dagger$	$3365^\dagger$	$3370^\dagger$	$3380^\dagger$
	Deterministic	$3315^\dagger$	$3302^\dagger$	$3305^\dagger$	$3313^\dagger$	$3338^\dagger$
Model-based	(not trained)	$2921^\dagger$	$3073^\dagger$	$2990^\dagger$	$2960^\dagger$	$3102^\dagger$

MAPPO with $\nu = 4$ stays at 3517–3557 from $N = 3$ to $N = 6$ and drops to 1843 and 1800 at $N = 7, 8$, where some runs fail (intervals down to 317). At $N = 2$ it is also lower (2783), consistent with Table 1. VDN is below MAPPO for $3 \leq N \leq 6$ but degrades more gradually (3316–3422 at $N \leq 4$, 2766–2812 at $N \in \{5, 6\}$, 2513 and 2097 at $N = 7, 8$), so it has the higher IQM at $N \in \{7, 8\}$. A full neighbor block ($\nu = N - 1$) does not help beyond the overlap of the intervals for either learner, so the decline at large N is not caused by robots failing to see their teammates. The IQM of every RL cell remains above the success threshold up to $N = 8$. The tuned model-based controller falls below it at $N = 5$ (960) and at $N \geq 7$ (468, 259). All of its failed episodes end in collisions (48–61% of the episodes at $N \in \{5, 7, 8\}$), none at the step limit.

Table 27: Wind-model ablation at $N = 3$, 5 seeds $\times$ 10 variations, 50 held-out episodes per run, 90% completion threshold: held-out IQM of policies trained under the row wind model and evaluated under the column wind model. AR(1): correlated wind of Eqs. 1–2 with $\tau_w = 20$ steps; i.i.d.: $\phi = 0$. The model-based controller is not trained, so its two rows are the same evaluation. †: IQM above 1530.

Trained under	**MAPPO** eval AR(1)	eval i.i.d.	**VDN** eval AR(1)	eval i.i.d.	**Model-based** eval AR(1)	eval i.i.d.
AR(1)	$3416^\dagger$ [3412, 3420]	$3418^\dagger$ [3414, 3422]	$3353^\dagger$ [3319, 3377]	$3371^\dagger$ [3334, 3388]	$2921^\dagger$ [2879, 2960]	$2953^\dagger$ [2898, 2992]
i.i.d.	$3416^\dagger$ [3407, 3422]	$3416^\dagger$ [3408, 3422]	$3353^\dagger$ [3308, 3379]	$3363^\dagger$ [3309, 3381]	$2921^\dagger$ [2879, 2960]	$2953^\dagger$ [2898, 2992]

Table 28: Scalability on variations 1, 5 and 9 (5 seeds each, 50 held-out episodes per run): held-out IQM with 95% stratified-bootstrap interval. RL learners use the hyperparameters tuned at $N = 3$. "—": not run, because $\nu = N - 1$ and $\nu = 4$ see the same teammates for $N \leq 5$. †: IQM above 1700.

Method	$N = 2$	$N = 3$	$N = 4$	$N = 5$	$N = 6$	$N = 7$	$N = 8$
MAPPO ($\nu = 4$)	$2783^{\dagger}$ [1860, 2937]	$3531^{\dagger}$ [3520, 3537]	$3557^{\dagger}$ [3547, 3563]	$3549^{\dagger}$ [3537, 3556]	$3517^{\dagger}$ [3412, 3541]	$1843^{\dagger}$ [631, 2662]	$1800^{\dagger}$ [317, 2930]
MAPPO ($\nu = N - 1$)	—	—	—	—	$3456^{\dagger}$ [3376, 3511]	$2059^{\dagger}$ [882, 3007]	$2246^{\dagger}$ [738, 2761]
VDN ($\nu = 4$)	$3330^{\dagger}$ [2714, 3411]	$3422^{\dagger}$ [2717, 3460]	$3316^{\dagger}$ [3236, 3422]	$2766^{\dagger}$ [2095, 3134]	$2812^{\dagger}$ [2461, 3006]	$2513^{\dagger}$ [2344, 2694]	$2097^{\dagger}$ [1118, 2553]
VDN ($\nu = N - 1$)	—	—	—	—	$2840^{\dagger}$ [2601, 2982]	$2234^{\dagger}$ [679, 2703]	$2276^{\dagger}$ [1952, 2491]
Model-based (tuned)	$2986^{\dagger}$ [2916, 3050]	$2671^{\dagger}$ [2588, 2834]	$2561^{\dagger}$ [2056, 2759]	960 [725, 1138]	$1887^{\dagger}$ [1543, 2096]	468 [289, 741]	259 [25, 493]

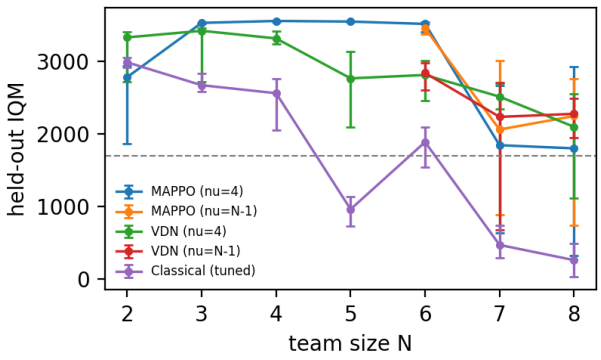


Figure 4: Scalability: held-out IQM with 95% intervals against team size on variations 1, 5 and 9, for the methods of Table 28. The dashed line is the success threshold of 1700.

E.6 Simulation Environment Details

We built a 3-D cornfield scene in CoppeliaSim (Rohmer et al., 2013) with weeds at randomized locations. CoppeliaSim was chosen for its real-time physics; rigid-body dynamics are computed by the PyBullet (Coumans & Bai, 2016–2021) engine, and our Gymnasium-compatible (Towers et al., 2024) environment drives it through the ZeroMQ (ZeroMQ, 2024) remote API. Field coordinates are mapped to simulator meters by a factor of 8, and the UAVs fly at a fixed altitude of 0.4 m, which is consistent with the planar navigation dynamics of the training environment: the altitude is held by a low-level controller that the formulation of Section 4 abstracts away.

Figs. 5 and 6 show all ten variations at $N = 3$ before and after policy execution.

Observation gap. Table 29 gives the full version of Table 3 of the main paper. The perturbations are applied to the observation at inference time only; the underlying dynamics are unchanged, so any degradation is attributable to the policy's use of corrupted inputs rather than to a harder task. Position noise is additive Gaussian noise of $\sigma = 5.0$ m on each robot's own position; it is physical for every method, i.e. it is divided by the position scale of the observation normalization before it reaches a normalized input. Velocity staleness replaces each robot's own velocity by its value 3 steps earlier. Field-map loss zeroes the entire infection map with probability 0.1 at each step. The combined condition applies all three with a 5-step lag. The perturbation targets the same semantic components for every observation type — each robot's own position and velocity and the shared infection map — so the columns are comparable. Under decentralized execution each robot's own sensors are corrupted independently, which is the Dec-POMDP counterpart of the centralized condition rather than a single shared corruption; the model-based controller receives the perturbed centralized observation through its observer.

Table 29: Observation-gap analysis ($N = 3$, tuned hyperparameters, 5 seeds $\times$ 10 variations, 100 held-out episodes per run and condition): held-out IQM with 95% stratified-bootstrap interval, mean per-run relative change Δ, and percentage of episodes that end sprayed. †: IQM above 1700.

Perturbation	MAPPO			VDN			Model-based		
	IQM	Δ (%)	Sprayed (%)	IQM	Δ (%)	Sprayed (%)	IQM	Δ (%)	Sprayed (%)
None (baseline)	$3532^{\dagger}$ [3527, 3536]	—	99.9	$3084^{\dagger}$ [2552, 3379]	—	83.6	$3009^{\dagger}$ [2972, 3038]	—	95.1
Position noise ($\sigma = 5.0$ m)	$3503^{\dagger}$ [3492, 3512]	−0.9	99.8	$2494^{\dagger}$ [1591, 2962]	−146.7	79.3	10 [−97, 117]	−98.9	47.7
Velocity staleness (3 steps)	$3528^{\dagger}$ [3523, 3532]	−0.1	99.9	$2631^{\dagger}$ [1820, 3016]	−176.2	79.0	$1787^{\dagger}$ [1720, 1848]	−46.7	84.7
Field-map loss (10%)	$3532^{\dagger}$ [3526, 3536]	+0.0	99.9	$2990^{\dagger}$ [2460, 3289]	−43.3	82.5	$3020^{\dagger}$ [2985, 3048]	−0.1	95.1
All combined (staleness 5)	$3475^{\dagger}$ [3461, 3488]	−1.9	99.5	1316 [51, 2064]	−329.8	70.9	−1261 [−1315, −1181]	−143.5	34.5

Table 30: Paired two-sided Wilcoxon signed-rank tests on the per-run relative drop of the observation-gap study ($N = 3$, pairs = (variation, seed) runs). Medians are of the per-run relative change (%).

Test	Condition	Pairs	Medians	W	p
MAPPO vs. VDN	Position noise	50	−0.50 / −10.26	89	3.4×10^{-9}
MAPPO vs. VDN	Velocity staleness	50	−0.07 / −9.38	127	7.8×10^{-8}
MAPPO vs. VDN	Field-map loss	50	−0.00 / −0.30	239	6.2×10^{-5}
MAPPO vs. VDN	All combined	50	−1.19 / −31.51	71	5.9×10^{-10}
MAPPO: position vs. staleness	—	50	−0.50 / −0.07	112	2.4×10^{-8}
VDN: position vs. staleness	—	50	−10.26 / −9.38	570	0.52

The study covers all 50 tuned $N = 3$ runs of MAPPO and VDN (5 seeds $\times$ 10 variations) and the tuned model-based controller on the same cells, with 100 held-out episodes per run and condition. Episode seeds and the per-episode corruption streams are shared by all conditions and all methods, so every comparison is paired. Δ is computed per run against that run's own unperturbed baseline and averaged over runs, with a stratified-bootstrap interval. Table 30 reports paired Wilcoxon signed-rank tests of the per-run relative drop, between MAPPO and VDN for each perturbation and, within each learner, between position noise and velocity staleness; the latter is also checked variation by variation. As a mechanistic diagnostic we record, for every MAPPO run, the mean absolute input gradient of the action with respect to the robot's own position (per m) and own velocity (per m/s) over on-policy states, i.e. how strongly the policy weights its kinematic inputs; VDN's argmax policy has no action gradient. MAPPO degrades by at most 1.9% (combined) and still completes 99.5% of the episodes under every condition. VDN degrades significantly more under every perturbation ($p \leq 6.2 \times 10^{-5}$, Table 30), with per-run median drops of 10.3% (position noise), 9.4% (staleness), 0.3% (map loss) and 31.5% (combined); its completion rate falls from 83.6% to 70.9% under the combined condition. The mean relative changes of VDN in Table 29 (-43% to -330%) are inflated by a few runs whose baseline return is near zero or negative, where a moderate absolute loss is a huge relative one (one run with a baseline of -85 contributes -7703%), and by a few runs that collapse under the perturbation. The medians and the IQMs are the better summaries. The model-based controller is the most fragile: position noise of 5 m removes almost all of its return (-98.9%, 47.7% of the episodes sprayed), since it aims every robot from its observed position. Staleness costs it 46.7%, and the combined condition drives its IQM to -1261. Map loss costs no method more than 0.3% in median, in line with the observation ablation. Within MAPPO, position noise hurts more than staleness (median drops 0.50% against 0.07%, $p = 2.4 \times 10^{-8}$), but both effects are below 1%. Within VDN, the two do not differ ($p = 0.52$). Position noise is the larger drop in all 10 variations for MAPPO, but in only 6 of the 10 for VDN. MAPPO's mean absolute action gradient is 0.063 per m of own position and 0.075 per m/s of own velocity, i.e. its actions depend only weakly on its own kinematic readings, which is consistent with its robustness.

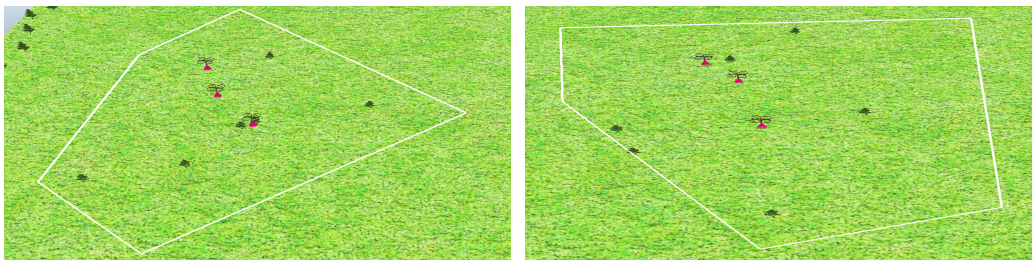
E.7 Reproducibility

The released code contains one entry point per stage of the pipeline. `src/env.py` implements the centralized POMDP of Section 5 (Eqs. 3–16, including the wind process of Eqs. 1–2), with its per-step physics in compiled kernels (`src/physics.py`), and `src/env_multi.py` the

Dec-POMDP wrapper, including the local observation of Eq. 21; the latter composes the former, so the two formulations provably share S, $\mathcal{T}$, R, b_0 and γ rather than merely being described as doing so. `src/marl/` holds the cooperative baselines behind a common interface; the centralized baselines come from Stable-Baselines3 unmodified. A single batched, seed-controlled evaluator (`src/evaluation.py`) produces every evaluation number, including the held-out seeds shared across methods.

`calibrate_spray.py` fits the spray kernel (Section C.3) and `feasibility_check.py` runs the feasibility check. `train.py` runs any algorithm in either formulation under any ablation condition, selected by command-line flag, ends every run with its held-out evaluation, and writes an identical log layout for both formulations. `tune.py` runs the Optuna studies, `classical_baseline.py` tunes and evaluates the model-based controller, and `sim2real.py` runs the observation-gap study and the CoppeliaSim bridge. `run_experiments.py` holds the complete experiment plan as a dependency graph of jobs: it runs the plan on a single machine and exports the same jobs as SLURM job lists, which the scripts in `slurm/` submit as dependent array jobs. `analyze_results.py` builds every table from the per-run logs and `plot_learning_curves.py` the learning curves. `exp_sets/stochastic_envs_v2.json` defines the ten variations of Table 6.

Supported but not reported. The released code also supports domain-randomization training and evaluation (in-distribution / out-of-distribution wind bands), a wind-speed sensitivity sweep, and a hyperparameter-sensitivity grid. They are not part of the results in this manuscript.



(a) Var. 1

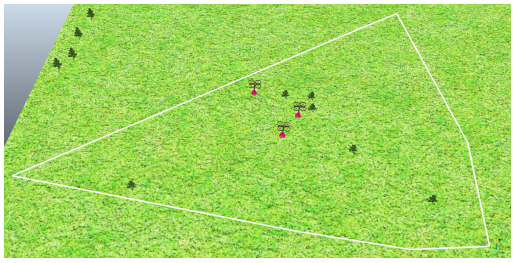
(b) Var. 2



(c) Var. 3



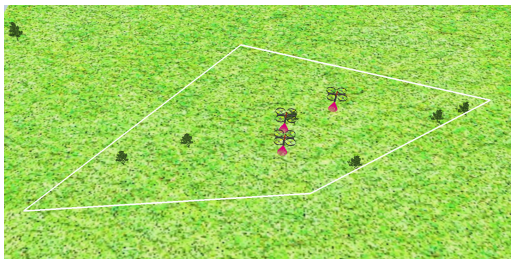
(d) Var. 4



(e) Var. 5



(f) Var. 6



(g) Var. 7



(h) Var. 8

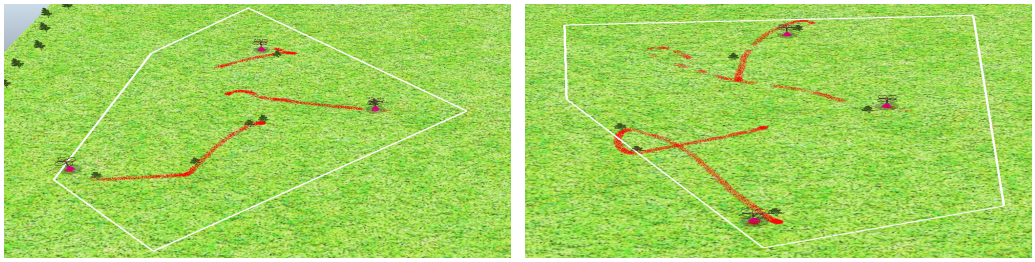


(i) Var. 9



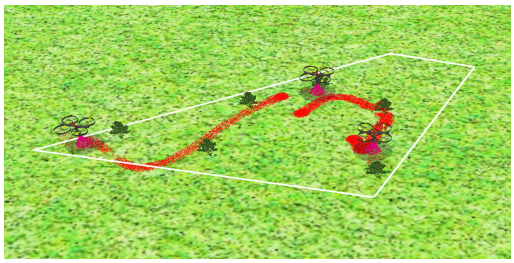
(j) Var. 10

Figure 5: Initial states of all ten environment variations in CoppeliaSim ($N = 3$). Dark green plants denote infection sites. Variations differ in field shape, launch positions and the placement of the six infection sites; see Table 6.

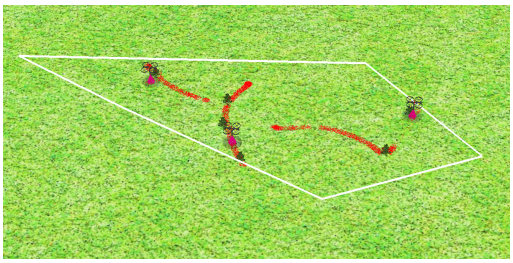


(a) Var. 1

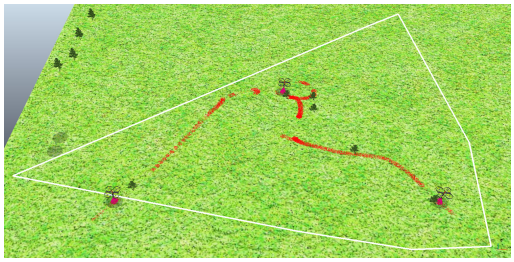
(b) Var. 2



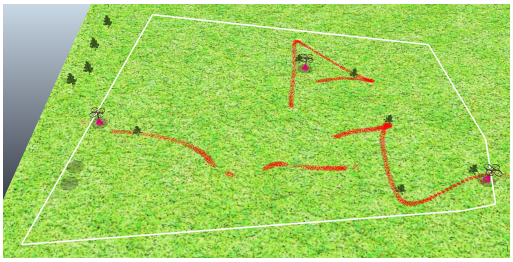
(c) Var. 3



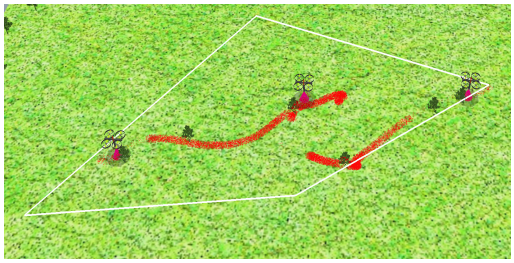
(d) Var. 4



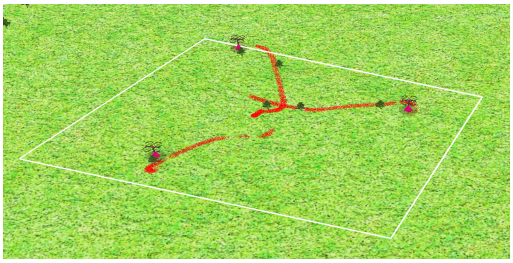
(e) Var. 5



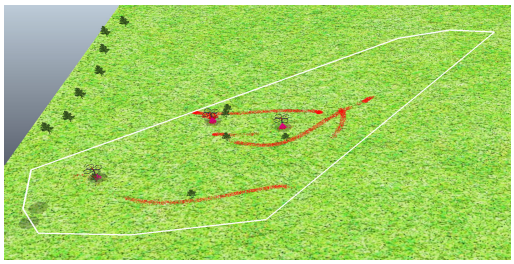
(f) Var. 6



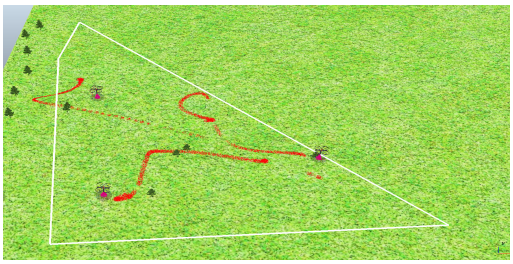
(g) Var. 7



(h) Var. 8



(i) Var. 9



(j) Var. 10

Figure 6: Final states of all ten variations after policy execution ($N = 3$). Red lines denote the UAV spraying paths.