Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

Adapting Execution-Time Objectives for Multi-Robot Policies via Collaborative Flow Policy Guidance

Williard Joshua Jose, Yuhao Li, Zihao Deng, and Hao Zhang
Human-Centered Robotics Lab, University of Massachusetts Amherst
{wjose, yuhaoli, zihaodeng, hao.zhang}@umass.edu

***Abstract*—Multi-robot teams are increasingly important for scaling task workload and complexity. However, existing approaches to multi-robot policy learning face three fundamental limitations: the inability of unimodal policies to capture multi-modal joint strategies, the rigidity of fixed policies in adapting to dynamic execution-time requirements, and the difficulty of resolving conflicting objectives during deployment. To address these challenges, we introduce *Collaborative Flow Policy Guidance* (CFPG), a novel approach for modifying collaborative multi-robot policies through objective composition at execution time. First, we propose Multi-Agent Flow Policy Optimization (MAFPO), which learns robust and multi-modal collaborative policies in a fully on-policy manner without requiring offline datasets. Second, we enable training-free multi-robot adaptation to new execution-time objectives by leveraging flow-matching guidance to steer actions of multi-robot systems toward user-specified goals. Third, we develop a hierarchical gradient projection mechanism that resolves conflicts between nominal objectives and execution-time objectives during guidance. We provide theoretical analysis of CFPG and validate its effectiveness across multiple simulation environments and real-world robot platforms. Experimental results demonstrate improved performance over state-of-the-art methods.**

Further details of this work are available on the project website: https://hcrlab.gitlab.io/project/cfpg.

I. Introduction

Collaborative multi-robot systems are becoming pivotal in robotics due to their ability to scale significantly in terms of task workloads in various applications, such as search and rescue [1], [2], warehouse logistics [3], [4], space exploration [5], [6], and environmental monitoring [7], [8]. By leveraging the capabilities of multiple robots, these systems can address large-scale challenges more efficiently than single-robot systems. However, as their scale and complexity increase, coordinating multi-robot actions to satisfy diverse, dynamic, and potentially conflicting objectives becomes increasingly challenging.

Existing approaches for multi-robot collaboration face three fundamental challenges, as illustrated in Fig. 1. First, classical Multi-Agent Reinforcement Learning (MARL) algorithms, such as MAPPO [9], typically model policies using unimodal Gaussian distributions. While sufficient for simple tasks, these distributions fail to capture the complex, multi-modal action distributions inherent in multi-robot collaboration, where valid strategies often lie in disjoint regions of the action space (e.g., passing either to the left or right of an obstacle to avoid collision). Consequently, these methods often converge to suboptimal or invalid mean actions when faced with multiple distinct solutions. The second challenge is that once trained, standard multi-agent policies are generally fixed and static.

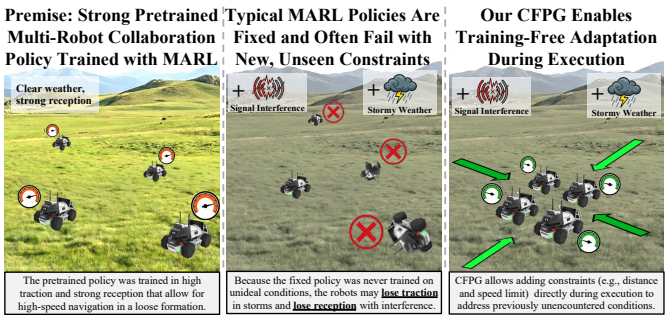

Fig. 1. A motivating scenario illustrating the challenges of balancing multiple potentially conflicting objectives in multi-robot teams. Our collaborative flow policy guidance explicitly enables execution-time learning-free multi-robot policy optimization across multiple goals and automatically resolves conflicts according to goal priorities.

Modifying them during execution to optimize new or varying objectives is challenging without retraining. This rigidity limits the team's ability to adapt to dynamic requirements that were not foreseen or prioritized during the initial training phase. The third challenge occurs when robot teams often must balance multiple competing objectives, such as maintaining formation versus avoiding obstacles. While techniques exist to handle objective conflicts during training (such as scalarization [10], [11] and gradient manipulation [12]–[14]), there is currently no straightforward mechanism to resolve these conflicts directly during execution time. As a result, deployed policies may fail to satisfy new objectives, especially when the robot team state and environmental conditions diverge from those encountered during initial training.

To address these challenges, we present *Collaborative Flow Policy Guidance* (CFPG), a novel and principled approach that enables training-free adaptation of pretrained collaborative multi-robot policies to new objectives directly during execution time. Unlike traditional MARL methods that fix reward preference weights during training, CFPG decouples the learning of collaboration strategies (CFPG training) from the satisfaction and balancing of new execution-time objectives (CFPG execution). Our CFPG method leverages a flow matching model trained via a novel algorithm we call *Multi-Agent Flow Policy Optimization* (MAFPO), which extends flow policy optimization in the single-agent case [15] to the multi-agent setting and represents multi-robot joint actions as a collaboration manifold. By adapting this approach to multi-

robot teams, MAFPO learns a robust, multi-modal distribution of valid joint actions in a fully on-policy manner, without relying on offline datasets or expert demonstrations. This generative representation provides a continuous probability density over the space of collaborative behaviors. At execution time, CFPG utilizes the guidance property of flow matching to steer the generated actions toward low-cost regions defined by user-specified differentiable cost functions. Furthermore, to explicitly handle situations where nominal and execution-time objectives pull in opposing and conflicting directions, we employ gradient projection techniques directly during the flow matching inference process to resolve the resulting gradient conflicts. We provide theoretical analysis and proofs of CFPG and validate our approach empirically.

The specific novelties of this work include:

- We introduce the first multi-agent flow matching policy trained in a fully on-policy manner without requiring any offline training dataset, and demonstrate its ability to learn diverse and robust collaborative policies with multi-modal action distributions successfully.
- We enable the training-free ability of adapting existing collaborative multi-robot policies to new objectives directly during execution time by using flow matching guidance.
- We propose a hierarchical gradient projection technique to resolve conflicts among different execution objectives, and between nominal (training) objectives and execution-time objectives.

II. Preliminaries

Multi-Agent Proximal Policy Optimization (MAPPO). Multi-robot collaboration can be modeled as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) defined by $\mathcal{M} = \langle \mathcal{I}, \mathcal{S}, \{\mathcal{A}_i\}_{i\in\mathcal{I}}, \mathcal{P}, \mathcal{R}, \{\mathcal{O}_i\}_{i\in\mathcal{I}}, \Omega, \gamma \rangle$, where $\mathcal{I}$ indexes the robots, $\mathcal{S}$ is the global state space, $\mathcal{A}_i$ and $\mathcal{O}_i$ are the action and observation spaces of robot i, and $\mathcal{A} = \prod_i \mathcal{A}_i$ is the joint action space. The transition function $\mathcal{P}$ defines the system dynamics, $\mathcal{R}$ is a shared reward function, Ω denotes the observation function, and $\gamma \in (0, 1]$ is the discount factor. At time t, $\mathbf{s}^t \in \mathcal{S}$; robots observe $\mathbf{o}_i^t = \Omega_i(\mathbf{s}^t) \in \mathcal{O}_i$, act via $\mathbf{a}_i^t \sim \pi_i(\cdot|\mathbf{o}_i^t)$, and induce $\mathbf{s}^{t+1} \sim \mathcal{P}(\cdot|\mathbf{s}^t, \mathbf{a}^t)$ with reward $r^t = \mathcal{R}(\mathbf{s}^t, \mathbf{a}^t)$. A joint policy π is learned by maximizing the expected discounted return $J(\pi) = \mathbb{E}_{\mathbf{a}^t\sim\pi}[\sum_{t=0}^{\infty} \gamma^t r^t]$. MAPPO [9] optimizes a clipped surrogate objective that approximates $J(\pi)$ while ensuring stable policy updates [16]:

$$\mathcal{L}^{\text{MAPPO}}(\theta) = \hat{\mathbb{E}}\left[\min(\rho(\theta)\hat{A}^t, \text{clip}(\rho(\theta), 1-\epsilon, 1+\epsilon)\hat{A}^t)\right] \tag{1}$$

where $\rho(\theta) = \frac{\pi_\theta(\mathbf{a}_i|\mathbf{o}_i)}{\pi_{\theta^{\text{old}}}(\mathbf{a}_i|\mathbf{o}_i)}$ is the probability ratio, and $\hat{A}^t$ is the Generalized Advantage Estimate (GAE) [17]. During training only, a separate critic network estimates the value function to help in computing advantage estimates.

Flow Matching. To model complex continuous action distributions for $\mathbf{a}$, we adopt flow matching [18], [19], an emerging generative modeling paradigm that learns a continuous-time vector field to transport a base distribution to the data distribution by directly supervising the trajectories of the flow.

A flow defines a probability path $p_\tau(\mathbf{a}, \tau) : \mathbb{R}^d \times [0, 1] \to \mathbb{R}_{\geq 0}$, which transforms a simple prior noise distribution $p_0(\mathbf{a}) = \mathcal{N}(\mathbf{a}; \mathbf{0}, \mathbf{I})$ (e.g., a Gaussian distribution at $\tau = 0$) to the data distribution $p_1(\mathbf{a})$ (i.e., at $\tau = 1$) via a time-dependent velocity field $v_\tau(\mathbf{a})$. The evolution of the probability density is governed by the continuity equation $\frac{\partial p_\tau}{\partial \tau} + \nabla \cdot (p_\tau v_\tau) = 0$, and samples can be generated by solving the Ordinary Differential Equation (ODE):

$$\frac{d\mathbf{a}}{d\tau} = v_\tau(\mathbf{a}), \quad \mathbf{a}_{\tau=0} \sim p_0 \tag{2}$$

Directly learning the probability density p_τ, as well as the marginal velocity field v_τ that generates p_τ, is intractable. Instead, flow matching leverages the concept of conditional flow matching (CFM), which assumes the marginal probability path is an aggregation of simpler conditional probability paths $p_\tau(\mathbf{a}|\mathbf{a}_{\tau=1})$ conditioned on a latent target data sample $\mathbf{a}_{\tau=1}$. In our setting, the entire generative process is additionally conditioned on a context variable $\mathbf{o}$ (e.g., robot observations). If $v_\tau(\mathbf{a}|\mathbf{a}_{\tau=1}, \mathbf{o})$ is the conditional velocity field generating $p_\tau(\mathbf{a}|\mathbf{a}_{\tau=1}, \mathbf{o})$, then the marginal velocity field $v_\tau(\mathbf{a}|\mathbf{o})$ satisfies $v_\tau(\mathbf{a}|\mathbf{o}) = \mathbb{E}_{p_\tau(\mathbf{a}_{\tau=1}|\mathbf{a},\mathbf{o})}[v_\tau(\mathbf{a}|\mathbf{a}_{\tau=1}, \mathbf{o})]$. This allows us to regress a neural network $v_\theta(\mathbf{a}, \tau|\mathbf{o})$ to the conditional velocity fields directly. We specifically employ an optimal transport path; given a context $\mathbf{o}$, a data sample $\mathbf{a}_{\tau=1} \sim q(\mathbf{a}|\mathbf{o})$, and a noise sample $\mathbf{a}_{\tau=0} \sim p_0$, we define the conditional path as a straight-line trajectory: $\mathbf{a}_\tau = (1-\tau)\mathbf{a}_{\tau=0} + \tau\mathbf{a}_{\tau=1}$. The target conditional velocity field generating this path is the time-derivative of the path, $v_\tau(\mathbf{a}_{\tau=0}, \mathbf{a}_{\tau=1}) = \mathbf{a}_{\tau=1} - \mathbf{a}_{\tau=0}$. Then, the CFM loss function is defined as:

$$\mathcal{L}^{\text{CFM}}(\theta) = \mathbb{E}_{\tau, \mathbf{a}_{\tau=0}, \mathbf{a}_{\tau=1}}\left[\|v_\theta(\mathbf{a}_\tau) - (\mathbf{a}_{\tau=1} - \mathbf{a}_{\tau=0})\|^2\right] \tag{3}$$

An advantage of flow matching is that it directly learns a deterministic transport vector field between noise and data, avoiding stochastic reverse diffusion and complex noise scheduling. Flow matching also performs faster model inference during execution, which aids in real-time robot deployment.

III. Problem Formulation

We consider a team of N robots operating within a shared physical environment in $SE(2)$. Each robot $i \in \{1, \ldots, N\}$ has pose $\mathbf{x}_i^t = [x_i^t, y_i^t, \theta_i^t]$ and velocity $\mathbf{v}_i^t = [v_{x,i}^t, v_{y,i}^t, \omega_i^t]$ at timestep t and has a radius of r^{robot}. The individual state is given by $\mathbf{s}_i^t = [\mathbf{x}_i^t, \mathbf{v}_i^t]$. Robots are controlled via actions $\mathbf{a}_i^t = [a_{x,i}^t, a_{y,i}^t, \alpha_i^t] \in \mathcal{A}_i$, representing linear and angular accelerations. The motion of each robot follows discrete-time dynamics defined by $\mathbf{v}_i^{t+1} = \mathbf{v}_i^t + \mathbf{a}_i^t \Delta t$ and $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^t \Delta t$, subject to kinematic constraints. To perceive the environment, each robot is equipped with a limited-range LIDAR sensor. The local observation $o_i^t \in \mathcal{O}_i$ includes the robot's pose information and a set of range readings $\{z_{i,k}^t\}_{k=1}^K$ capturing the distance to obstacles or other robots within a sensing radius r^{lidar}.

The global state of the collaborative robot team, denoted by $\mathbf{s}^t = [\mathbf{x}_1^t, \ldots, \mathbf{x}_N^t] \in \mathcal{S}$, represents the full configuration

of all robots. The observations $\mathbf{o}_i^t$ only provide partial and noisy information about this global state. We define collision states as configurations where the spatial distance between any two robots i, j satisfies $\|\mathbf{x}_i^t, \mathbf{x}_j^t\|_2 \leq 2r^{\text{robot}}$. The joint action $\mathbf{a}^t = [\mathbf{a}_1^t, \ldots, \mathbf{a}_N^t]$ transitions the system state according to the joint dynamics $\mathcal{P}(\mathbf{s}^{t+1}|\mathbf{s}^t, \mathbf{a}^t)$.

The robot team must collaborate to satisfy a set of objectives. First, we define the **Nominal Team Objective** (J^{nom}), which represents the fundamental objective of the robot team; this can be a standard multi-robot task such as collision-free multi-robot navigation. We formulate this objective by identifying a set of optimal joint actions $\mathcal{A}^{\text{nom}}(\mathbf{s}^t)$ at each state that minimize the task cost $J^{\text{nom}}(\mathbf{s}^t, \mathbf{a}^t)$ as: $\mathcal{A}^{\text{nom}}(\mathbf{s}^t) = \text{argmin}_{\mathbf{a}^t} J^{\text{nom}}(\mathbf{s}^t, \mathbf{a}^t)$.

In addition to the nominal task, we consider a set of **Dynamic Execution Objectives** (J^{exec}). These are defined by M cost functions $\{J_m^{\text{exec}}(\mathbf{a}^t)\}_{m=1}^M$. At execution time, these are aggregated into a single scalar cost $\sum_{m=1}^M \lambda_m J_m^{\text{exec}}(\mathbf{a}^t)$, parameterized by a preference vector $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_M\} \in \mathbb{R}^M$. A concrete example of a dynamic execution objective is controlling the joint motion of the robots in the team to have velocity magnitudes within a specified range (i.e., fast and efficient vs slow and careful movement). Another example is specifying the relative positions of robots during collaboration (i.e., robot 1 must be to the left of robot 2, and robot 3 must be to the right of robot 2). The problem we want to solve is how to optimize these execution objectives strictly within the manifold of behaviors that satisfy the nominal objective J^{nom}. This can be formulated as a bilevel optimization problem:

$$\mathcal{A}^{\text{exec}} = \underset{\tilde{\mathbf{a}}^t \in \mathcal{A}^{\text{nom}}}{\text{argmin}} \sum_{m=1}^{M} \lambda_m J_m^{\text{exec}}(\tilde{\mathbf{a}}^t) \tag{4}$$

$$\text{s.t.} \quad \mathcal{A}^{\text{nom}} = \underset{\mathbf{a}^t \in \mathcal{A}}{\text{argmin}}\, J^{\text{nom}}(\mathbf{s}^t, \mathbf{a}^t) \tag{5}$$

Here, the lower-level optimization defines the feasible set of task-optimal actions, from which the upper-level optimization selects the action that best aligns with $\boldsymbol{\lambda}$. This bilevel structure highlights the intractability: the constraint set $\mathcal{A}^{\text{nom}}(\mathbf{s}^t)$ is defined by the solution to a complex, non-convex optimization problem (often involving long-horizon value estimation) in Eq. (5) that cannot be solved analytically online and may consist of disjoint, multi-modal manifolds. Furthermore, this set must be further optimized with additional dynamic execution-time objectives in Eq. (4) during execution time, further exacerbating the tractability challenge.

To overcome the intractability of the bilevel optimization problem, we propose to first learn a conditional generative policy that approximates the distribution over the optimal set $\mathcal{A}^{\text{nom}}(\mathbf{s}^t)$, from which we will sample actions that optimize Eq. (4). To realize this approach, we require two components: (1) a method to train this policy to maximize a pretraining reward that optimizes J^{nom}, which we formulate as a Multi-Agent Reinforcement Learning (MARL) problem, and (2) a generative model capable of representing the complex, multi-modal policy that can be sampled while optimizing J^{exec}, for which we employ flow matching. We reformulate the execution-time optimization problem in Eqs. (4)-(5) by sampling from the action distribution represented by a learned policy:

$$\mathbf{a}^* = \underset{\tilde{\mathbf{a}}^t \in \mathcal{A}_\varepsilon^{\text{nom}}}{\text{argmin}} \sum_{m=1}^{M} \lambda_m J_m^{\text{exec}}(\tilde{\mathbf{a}}^t) \tag{6}$$

$$\text{s.t.} \quad \mathcal{A}_\varepsilon^{\text{nom}} = \{\pi \mid G(\pi) \geq G(\pi^*) - \varepsilon\} \tag{7}$$

where $G(\pi) = \mathbb{E}_{\mathbf{s}^t, \mathbf{a}^t \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(\mathbf{s}^t, \mathbf{a}^t)\right]$ is the expected total return of a policy π, and π^* is the optimal policy maximizing this return (representing J^{nom}). The nominal objective J^{nom} encodes the full task objective for which a reward $\mathcal{R}$ is defined and directly optimized during training. In contrast, J^{exec} encodes additional execution-time objectives on top of the policy trained on $\mathcal{R}$. By utilizing this formulation, we minimize over the set of all ε-optimal policies, allowing us to satisfy the complex pretraining constraints implicitly via the generative model while explicitly optimizing the dynamic execution objectives. The approach consists of: (1) an on-policy training algorithm to learn the collaborative manifold to solve Eq. (7), and (2) an execution-time guidance mechanism utilizing gradient projection to solve Eq. (6).

Eq. (7) can be solved by MAPPO to get an unguided policy π_θ^* that maximizes the expected discounted reward. However, while effective for unimodal Gaussian policies, MAPPO struggles to represent the multi-modal action distributions that arise in complex multi-robot interactions, where feasible actions often lie in disjoint regions of the action space. Furthermore, although gradient projection techniques such as PCGrad [12] could theoretically be employed to balance conflicting rewards during MAPPO training (though not well explored in literature), they cannot be directly applied to resolve conflicts during execution. Once trained, MAPPO policies are fixed and generally lack the ability to adapt actions to new objectives on the fly.

IV. Approach

We propose *Collaborative Flow Policy Guidance* (CFPG), a principled approach that learns a robust multi-robot collaboration policy, which can then be used to optimize specific and potentially conflicting execution-time objectives. CFPG consists of learning a static policy $\pi_\theta(\mathbf{a}|\mathbf{o})$ that approximates the manifold of valid collaborative joint actions and modifying it during execution time to optimize the additional objectives. By introducing a new MARL approach called Multi-Agent Flow Policy Optimization (MAFPO), we train a flow matching model as the policy backbone that can be modified during model inference. CFPG leverages MAFPO's learned velocity field v_θ to model the distribution of robot actions as a prerequisite for additional optimization during guided generation. An overview of the CFPG architecture is illustrated in Fig. 2. This formulation treats the policy as a learnable velocity field, which can be continuously steered during the inference process (ODE integration) to minimize dynamic costs via gradient guidance, while employing gradient projection techniques to resolve conflicts between team-level and individual-level objectives.

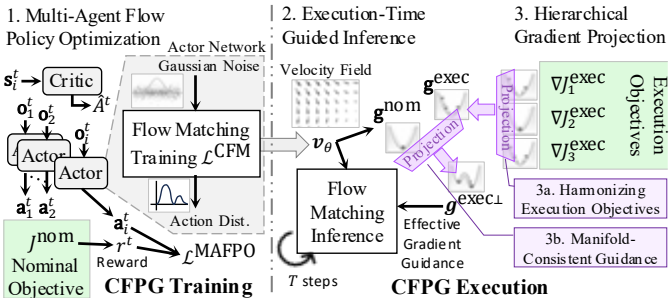

Fig. 2. CFPG trains a flow matching model within the MARL framework using our novel multi-agent flow policy optimization. During execution, CFPG enables training-free guided inference to optimize new objectives, modifying the actions from the pretrained policy. To resolve conflicts among objectives, hierarchical gradient projection is applied, first across execution objectives, and then between the combined execution objective and the nominal objective.

A. Multi-Agent Flow Policy Optimization (MAFPO)

One of our primary novelties is Multi-Agent Flow Policy Optimization (MAFPO), which explicitly addresses the theoretical limitations of unimodal action distributions in classical MARL. By integrating flow matching into the MARL paradigm, MAFPO learns decentralized flow-based policies π_{θ_i} for each robot while utilizing a centralized value function $V_\phi(\mathbf{s})$ to estimate advantages $\hat{A}^t$ based on the global state [9]. In contrast to standard flow matching policies that rely on behavior cloning from offline datasets, MAFPO is trained in a fully on-policy manner, enabling the learning of robust, multi-modal collaborative strategies without expert demonstrations. We extend Flow Policy Optimization (FPO) [15] to multi-robot collaboration, and similarly reframe policy optimization by using the CFM objective as a surrogate for log-likelihood. Specifically, we replace the probability ratio $\rho(\theta) = \frac{\pi_\theta(\mathbf{a}^t|\mathbf{o}^t)}{\pi_{\theta^{\text{old}}}(\mathbf{a}^t|\mathbf{o}^t)}$ from MAPPO using the difference in flow matching losses:

$$\rho^{\text{MAFPO}}(\theta) = \exp\left(\mathcal{L}^{\text{CFM}}(\theta^{\text{old}}; \mathbf{a}^t, \mathbf{o}^t) - \mathcal{L}^{\text{CFM}}(\theta; \mathbf{a}^t, \mathbf{o}^t)\right) \quad (8)$$

where $\mathcal{L}^{\text{CFM}}$ is the flow matching loss evaluated at the sampled action, and θ^{old} are the weights of the policy in the previous iteration.

While this ratio is used in the standard PPO-clip objective, the clipping mechanism can still diverge and may also be restrictive for the complex, multi-modal distributions learned in multi-robot collaboration. To address this, we adopt Simple Policy Optimization (SPO) [20], which replaces hard clipping with a soft Kullback-Leibler (KL) divergence regularization. We generalize SPO to the multi-robot setting by optimizing the joint policy via individual updates. The resulting MAFPO objective for robot i is:

$$\mathcal{L}^{\text{MAFPO}}(\theta_i) = -\hat{\mathbb{E}}\left[\rho^{\text{MAFPO}}(\theta_i)\hat{A}^t - \frac{|\hat{A}^t|}{2\epsilon}[\rho^{\text{MAFPO}}(\theta_i) - 1]^2\right] \quad (9)$$

where the probability ratio $\rho^{\text{MAFPO}}(\theta_i)$ is computed per robot i using the flow matching loss difference $\mathcal{L}^{\text{CFM}}(\theta_i^{\text{old}}) - \mathcal{L}^{\text{CFM}}(\theta_i)$. This formulation allows MAFPO to learn robust, multi-modal collaborative policies in a stable manner without requiring offline datasets or expert demonstrations. By independently using local observations and learning individual action multimodality, the robots implicitly learn coherent multimodal joint actions.

Proposition 1 (MAFPO Surrogate Consistency)**.** *Minimizing the MAFPO surrogate objective $\mathcal{L}^{MAFPO}(\theta_i)$ yields a policy update direction that maximizes the advantage-weighted Evidence Lower Bound (ELBO). Specifically, $\nabla_{\theta_i}\mathcal{L}^{MAFPO} = -\mathbb{E}[\hat{A}^t \nabla_{\theta_i} ELBO(\mathbf{a}_i|\mathbf{o}_i)]$. Because the ELBO is a variational lower bound on the log-likelihood, this update approximately maximizes the expected joint return $J(\boldsymbol{\theta})$.*

Proof. See the Supplementary Material.

B. Execution-Time Guided Inference

In order to solve the challenge of optimizing new execution-time objectives, we propose to use guidance on the trained MAFPO policies. While guidance has been explored for single-agent diffusion, we extend this to multi-agent flow policies, enabling the novel capability of modifying collaborative strategies on-the-fly without retraining. Once trained, π_θ represents the distribution of valid collaborative actions that optimize J^{nom}. At execution time, we aim to steer this distribution towards actions that minimize the dynamic execution costs defined by $J^{\text{exec}}(\mathbf{a}) = \sum_{m=1}^{M} \lambda_m J_m^{\text{exec}}(\mathbf{a})$, where $\boldsymbol{\lambda} \in \mathbb{R}^M$ is the preference vector.

We leverage the flexibility of flow matching to modify the generation process by introducing a guidance term to the velocity field that steers the flow toward low-cost regions. The guided ordinary differential equation (ODE) is given by:

$$\frac{d\mathbf{a}_\tau}{d\tau} = v_\theta(\mathbf{a}_\tau) - \eta \nabla_{\mathbf{a}_\tau} J^{\text{exec}}(\hat{a}(\mathbf{a}_\tau, \tau)) \quad (10)$$

where $\hat{a}$ is an estimator function of $\mathbf{a}_{\tau=1}$ given by $\hat{a}(\mathbf{a}_\tau, \tau) = \mathbf{a}_\tau + (1-\tau)v_\theta(\mathbf{a}_\tau, \tau|\mathbf{o})$ under the optimal transport path assumption employed in MAFPO. Note that the optimal joint actions are conditioned on the local observations $\mathbf{o}$. Here, η is the guidance scale, which can also subsume the magnitude of the preference vector $\boldsymbol{\lambda}$ within J^{exec}. The input to the cost function is $\hat{a}(\mathbf{a}_\tau, \tau)$ because the execution costs are defined on the action distribution at $\tau = 1$ (i.e., $\mathbf{a}_{\tau=1}$), which corresponds to valid physical control commands. On the other hand, the intermediate states $\mathbf{a}_\tau$ are noisy interpolations that lack physical semantic meaning; evaluating costs directly on $\mathbf{a}_\tau$ would be ill-defined. Therefore, we must estimate the terminal action from the current state $\mathbf{a}_\tau$ to evaluate the cost. The guidance gradient is then computed via chain rule through the estimator $\hat{a}$:

$$\nabla_{\mathbf{a}_\tau} J^{\text{exec}}(\hat{a}(\mathbf{a}_\tau, \tau)) = \left(\nabla_{\mathbf{a}_\tau} \hat{a}(\mathbf{a}_\tau)\right)^\top \nabla_{\hat{\mathbf{a}}_{\tau=1}} J^{\text{exec}}(\hat{a}(\mathbf{a}_\tau, \tau)) \quad (11)$$

Inspired by other theoretical works [21], [22] on diffusion and flow matching, we prove a theorem on the execution-time guided flow regularized optimization for multi-robot systems. This guidance mechanism ensures that the generated actions approximately optimize the execution-time objectives using its

gradients while being regularized by the underlying probability flow during the ODE solution steps.

Theorem 1 (Guided Flow Regularized Optimization)**.** *The guided velocity field, defined by:*

$$\tilde{v}_\tau = v_\theta(\mathbf{a}_\tau, \tau) - \eta \nabla_{\mathbf{a}_\tau} J^{exec}(\hat{a}(\mathbf{a}_\tau, \tau)) \tag{12}$$

generates trajectories that approximate the descent direction of a time-dependent regularized objective:

$$\mathcal{L}_\tau(\mathbf{a}_\tau) = J^{exec}(\hat{a}(\mathbf{a}_\tau, \tau)) - \frac{1}{\eta} \log p_\tau(\mathbf{a}_\tau) \tag{13}$$

where p_τ represents the flow-induced marginal density and η is the guidance scale.

Proof. See the Supplementary Material.

Theorem 1 implies that the guidance optimizes the execution cost projected from the terminal state, regularized by the likelihood of the collaborative manifold. This formulation identifies $\tilde{v}_\tau$ as the negative gradient flow of the potential function $\mathcal{L}_\tau(\mathbf{a}_\tau)$. The term $J^{\text{exec}}(\hat{a}(\mathbf{a}_\tau, \tau))$ drives the system toward states that map to low-cost terminal actions, while the term $-\frac{1}{\eta} \log p_\tau(\mathbf{a}_\tau)$ acts as a regularizer. This regularization penalizes deviations from the high-likelihood regions of the learned collaborative policy, ensuring that the optimization of J^{exec} remains on the valid collaboration manifold.

C. Conflict Resolution via Hierarchical Gradient Projection

We aim to apply guided inference over multiple execution-time objective functions defined by $\{J_m^{\text{exec}}\}_{m=1}^M$. A fundamental challenge in multi-objective guidance is the occurrence of **gradient conflicts**, where the optimization direction of one objective opposes that of another. In our approach, such conflicts arise at two distinct levels: (1) among the execution-time objectives, and (2) between the aggregated execution guidance J^{exec} and the nominal objective J^{nom}. Naive summation of corresponding gradients can lead to destructive interference, resulting in trajectories that either violate team collaboration or fail to satisfy individual constraints. To robustly handle these conflicts, we propose a hierarchical gradient projection mechanism inspired by PCGrad [12] and FCGrad [14].

1) Harmonizing Execution Objectives: We resolve conflicts among the M execution objectives to ensure coherent guidance. Let $\mathcal{G} = \{\mathbf{g}_m\}_{m=1}^M$ be the set of descent directions for each execution objective, where $\mathbf{g}_m = -\nabla_{\mathbf{a}_\tau} J_m^{\text{exec}}(\hat{a}(\mathbf{a}_\tau, \tau))$. We seek a modified set of gradients $\{\tilde{\mathbf{g}}_m\}_{m=1}^M$ that minimizes destructive interference. For each gradient $\mathbf{g}_m$, we iteratively project it onto the normal plane of any conflicting gradient $\mathbf{g}_n$ (where $\mathbf{g}_m^\top \mathbf{g}_n < 0$) sampled from $\mathcal{G}$:

$$\tilde{\mathbf{g}}_m = \mathbf{g}_m - \frac{\mathbf{g}_m^\top \mathbf{g}_n}{\|\mathbf{g}_n\|^2} \mathbf{g}_n \quad \forall \mathbf{g}_n \text{ s.t. } \mathbf{g}_m^\top \mathbf{g}_n < 0 \tag{14}$$

This procedure harmonizes the execution objectives, yielding a consolidated guidance vector $\mathbf{g}^{\text{exec}} = \sum_{m=1}^M \lambda_m \tilde{\mathbf{g}}_m$.

Proposition 2 (Pareto Descent of Execution Objectives)**.** *The harmonized execution gradient $\mathbf{g}^{exec}$ derived through iterative projection lies within the intersection of the descent cones of the individual objectives $\{J_m^{exec}\}_{m=1}^M$, provided such an intersection exists. Consequently, an update along $\mathbf{g}^{exec}$ does not increase any individual cost J_m^{exec} locally.*

Proof. See the Supplementary Material.

This mechanism ensures that the optimization process does not trade off one objective for another but rather seeks a solution that respects all constraints simultaneously. Intuitively, if one objective requires moving *forward* and another *right*, the harmonized gradient will point *forward-right*. If they require *forward* and *backward*, the projection will find a neutral direction (i.e., in this case, no motion) rather than oscillating, thereby preventing an increase in the cost of individual objectives.

2) Manifold-Consistent Guidance: Next, we ensure that the execution guidance does not degrade the nominal collaborative policy. The learned velocity field $v_\theta(\mathbf{a}_\tau, \tau)$ implicitly encodes the manifold of valid collaborative actions (which follows J^{nom}). To strictly prioritize the learned collaboration structure, we define the nominal gradient $\mathbf{g}^{\text{nom}} = v_\theta(\mathbf{a}_\tau, \tau)$ as the primary objective. We assess the alignment between the nominal gradient and the guidance using their inner product $\rho = \mathbf{g}^{\text{exec}\top} \mathbf{g}^{\text{nom}}$. If $\rho < 0$, indicating a conflict where the guidance opposes the collaborative flow, we project $\mathbf{g}^{\text{exec}}$ onto the orthogonal complement of $\mathbf{g}^{\text{nom}}$:

$$\mathbf{g}^{\text{exec}\perp} = \mathbf{g}^{\text{exec}} - \frac{\mathbf{g}^{\text{exec}\top} \mathbf{g}^{\text{nom}}}{\|\mathbf{g}^{\text{nom}}\|^2} \mathbf{g}^{\text{nom}} \tag{15}$$

This projection ensures that the guidance term preserves team collaboration while locally optimizing dynamic constraints.

Proposition 3 (Non-Destructive Projection of Execution Guidance to the Collaborative Manifold)**.** *Let $\mathbf{g}^{nom} = v_\theta(\mathbf{a}_\tau, \tau)$ be the nominal velocity field and $\mathbf{g}^{exec}$ be the harmonized execution descent direction. The effective update direction $\mathbf{d} = \frac{d\mathbf{a}_\tau}{d\tau} = \mathbf{g}^{nom} + \mathbf{g}^{exec\perp}$, where $\mathbf{g}^{exec\perp}$ is the projection of $\mathbf{g}^{exec}$ with respect to $\mathbf{g}^{nom}$, satisfies $\langle \mathbf{d}, \mathbf{g}^{nom} \rangle \geq \|\mathbf{g}^{nom}\|^2$.*

Proof. See the Supplementary Material.

A key theoretical novelty of our approach is the guarantee that execution-time interventions are non-destructive with respect to the learned collaboration manifold. Unlike naive addition of gradients which can disrupt the delicate collaboration learned by the pretrained policy, our projection ensures that any modification aligns with the original nominal objective represented by the velocity field. Proposition 3 shows the guarantee that the execution-time optimization never reduces the alignment with the nominal collaborative flow, thereby preserving the learned coordination structure. It also shows that the guidance can operate strictly in the null space of the nominal velocity field as necessary.

V. Experiments

A. Experimental Setup

We train and evaluate N omnidirectional robots in a 2D physics-based environment, where each robot is controlled via continuous 2D force vectors. Across multiple environments, we

TABLE I
OVERVIEW OF VMAS ENVIRONMENTS, OBJECTIVES, AND EVALUATION METRICS.

Environment	Objective	Metric (unit)
Navigation	**N**: Navigation	Navigation Success (%)
	E: Speed Limit	Speeding Violation (%)
	E: Safety Margin	Margin Violation (%)
Balance	**N**: Payload Transportation	Payload Distance (m)
	E: Spatial Order	Order Violation (%)
	E: Inter-robot spacing	Spacing Stddev (m)
Sampling	**N**: Sample Gathering	Mean Samples (#)
	E: Geofencing	Geofencing Violation (%)
	E: Clustering	Cluster RMSE (m)

N: nominal objective; **E**: execution objective.

define two types of objectives: a **nominal objective**, optimized during training to learn a base policy, and two **execution objectives**, which represent novel goals introduced only at test time via the cost functions J^{exec} (Section IV-B). Performance is evaluated based on whether the nominal objective is satisfied *simultaneously* with both execution objectives, requiring methods to jointly optimize all three objectives during execution. We describe the baselines, environments, objectives, and evaluation metrics below, with additional experimental details provided in the Supplementary Material.

Baselines. To validate the efficacy of CFPG, we compare against baseline approaches that use MAPPO to train a base policy; however, these policies are typically already fixed during execution and cannot be modified. To equip these baselines with the ability to handle execution objectives unseen during training, we integrate them with *projected gradient descent* (PGD) [23], [24] and *shielding* (SHLD) [25], [26], two techniques primarily used in other related fields, such as adversarial machine learning and safe reinforcement learning, respectively, to post-process action predictions. **MAPPO+PGD** applies PGD by projecting the MAPPO-predicted action to its local neighborhood that minimizes execution costs J^{exec} using gradient descent. **MAPPO+SHLD** restricts the MAPPO predicted action onto a safe set to strictly enforce safety requirements. For this baseline, we repurpose shielding safety constraints to instead adapt to execution objectives. To further highlight the fundamental difference in our problem setting, we also introduce **PC-MAPPO**, a preference-conditioned multi-objective RL algorithm adapted to MAPPO. Unlike CFPG which addresses training-free execution-time adaptation to previously unseen constraints, PC-MAPPO requires the execution objectives to be pre-defined and optimized *during the training phase*. Finally, we include an ablation of our method, **CFPG (No HGP)**, which removes the Hierarchical Gradient Projection to isolate its effect on resolving conflicts between the nominal and execution objectives.

Environments, Objectives, and Metrics. We evaluate performance across three challenging environments implemented in the Vectorized Multi-Agent Simulator (VMAS) [27]. Each environment is defined by one nominal objective that specifies the primary task to be accomplished, along with two execution objectives that introduce additional behavioral constraints or preferences during execution. To assess the flexibility and adaptability of the methods, we consider both individual-level execution objectives, which apply to each agent separately, and team-level execution objectives, which require coordinated group behavior. For each environment, we further design objective-specific evaluation metrics that directly quantify the ability of different methods to optimize the corresponding objectives, enabling a fine-grained comparison across distinct behavioral criteria. Table I provides an overview of the correspondence between environments, nominal and execution objectives, and their associated evaluation metrics.

`Navigation`. The nominal objective is the *navigation* of N robots to N corresponding goal locations while avoiding collisions within a time limit. Its *navigation success* metric is the percentage of episodes where all robots reach their assigned goals without any inter-robot collisions. We define two execution objectives and their corresponding metrics. (1) *Speed limit* (individual) restricts each robot's velocity as a safety constraint. Its *speeding violation* metric measures the percentage of time steps a robot's velocity exceeds a predefined threshold. (2) *Safety margin* (team) requires a minimum inter-robot separation distance as an additional safety constraint. Its *margin failure* metric measures the percentage of time steps where inter-robot distance is below a predefined threshold.

`Balance`. The nominal objective is the *payload transportation* of N robots to a target zone within a time limit; gravity acts on the payload, requiring constant coordinated force for balance. Its *payload distance* metric is the mean distance of the payload (meters) from the target location. We also define two execution objectives with corresponding metrics. (1) *Spatial order* (individual) constrains the robot formation to a specific x-axis ordering (e.g., $x_1 < x_2 < x_3 < x_4$) to facilitate scanning. Its *order violation* metric measures the percentage of episodes where the predefined spatial ordering of robots is violated. (2) *Inter-robot spacing* (team) constrains the distance between a robot and its closest neighbor. Its *spacing stddev* metric measures the standard deviation (meters) of the inter-robot distances at the end of all episodes.

`Sampling`. The nominal objective is the *sample gathering* of N robots within an environment and within a time limit; the samples are spatially distributed with three distinct peaks. Its *mean samples* metric is the average number of samples gathered by the robot team. Two execution objectives are defined with corresponding metrics. (1) *Geofencing* (individual) prohibits robots to enter an "avoid" zone. Its *geofencing violation* metric measures the percentage of time steps any robot is within the avoid zone. (2) *Clustering* (team) requires robots to stay within a defined radius of the team centroid to facilitate flocking. Its *cluster RMSE* metric measures the root mean square error (meters) between the team's actual cluster radius and a target cluster radius. Here, the cluster radius is defined as the average distance of each robot to the team centroid.

Multi-Objective Success Rate. Since execution objectives may conflict with each other and the nominal objective, the goal is for each method to find a balance that satisfies all objectives to the extent possible. To clearly facilitate comparison among the

TABLE II
QUANTITATIVE COMPARISON OF EXECUTION-TIME ADAPTATION METRICS ACROSS THREE MULTI-ROBOT ENVIRONMENTS.

Method	Navigation			Balance			Sampling		
	Navigation Success (%) ↑	Speeding Violation (%) ↓	Margin Violation (%) ↓	Payload Dist. (cm) ↓	Order Violation (%) ↓	Spacing StdDev (cm) ↓	Mean Samples ↑	Geofencing Violation (%) ↓	Cluster RMSE (cm) ↓
MAPPO+PGD	89.70±1.15	0.03±0.01	33.54±1.43	82.19±4.78	67.30±2.45	26.58±3.62	**124.37±0.93**	33.13±2.31	14.62±0.24
MAPPO+SHLD	47.90±2.68	**0.00±0.00**	**1.30±0.07**	121.93±3.22	47.40±1.21	**12.49±0.24**	90.42±1.45	14.20±0.38	9.08±0.34
CFPG (Ours)	**92.90±2.01**	**0.00±0.00**	4.54±0.19	**60.77±2.27**	**25.06±2.49**	19.05±1.17	98.51±2.41	**13.36±1.44**	**8.64±0.32**

Shaded and clear cells indicate metrics for nominal and execution objectives, respectively. ↑ higher is better, ↓ lower is better, best results in bold.

TABLE III
MULTI-OBJECTIVE SUCCESS RATE ACROSS ENVIRONMENTS.

Method	**Navigation** (%) ↑	**Balance** (%) ↑	**Sampling** (%) ↑
MAPPO+PGD	59.64±0.82	1.30±0.97	50.80±3.35
MAPPO+SHLD	47.32±2.62	3.60±0.42	60.20±2.41
PC-MAPPO	76.24±1.10	0.00±0.00	24.50±3.00
CFPG (No HGP)	**89.14±1.13**	35.10±3.15	61.20±1.86
CFPG (Ours)	88.71±1.82	**38.20±2.68**	**71.00±0.94**

methods across all objectives, we define a unified metric for overall success: the *multi-objective success rate*. This metric measures the percentage of episodes where *all of the objectives*, including the nominal objective and all execution objectives, are simultaneously satisfied. Consequently, a high multi-objective success rate indicates a method's ability to effectively balance all objectives concurrently.

B. Results on Vectorized Multi-Agent Simulator

Quantitative Results and Statistical Significance. We present quantitative results in Table II reporting standard deviations over 5 seeds. In the Navigation environment, CFPG achieves the highest navigation success rate and ties for the best performance in minimizing speeding violations. While MAPPO+SHLD achieves a lower margin violation rate compared to CFPG, it results in a much lower navigation success rate, likely due to the overly strict enforcement of safety constraints. Similarly, in Balance, CFPG achieves the best payload distance and spatial order violation. In Sampling, CFPG achieves the lowest geofencing violation rate and cluster RMSE.

Conflict Resolution among Nominal and Execution Objectives. We further evaluate the ability of the methods to satisfy all three objectives simultaneously using the multi-objective success rate, as shown in Table III. Here, it is much clearer that CFPG significantly outperforms both baselines across all environments. Of particular note is the Balance environment, where achieving the specific spatial ordering of robots is an extremely challenging task to complete during execution time alone. This difficulty leads to very low success rates for the MAPPO+PGD and MAPPO+SHLD baselines. In contrast, CFPG achieves a significantly higher success rate, demonstrating its superior ability to balance and satisfy multiple conflicting objectives during execution. Using Welch's t-test, CFPG significantly outperforms baselines (MAPPO+PGD, MAPPO+SHLD, PC-MAPPO) in multi-objective success rate across Navigation (p=[8.8e-7, 6.9e-8, 3.5e-5]), Balance (p=[5.2e-6, 3.5e-5, 3.5e-5]), and Sampling (p=[4.9e-4, 1.2e-3, 4.8e-6]). In our ablation, CFPG outperforms [No HGP] in Balance (p=6.67e-2) and Sampling (p=2.38e-5) while maintaining equivalent performance in Navigation using Two One-Sided Tests (TOST) for equivalence (p=1.15e-3).

Execution-Time Guided Inference. We qualitatively demonstrate CFPG's adaptation to execution-time objectives in Fig. 3. In Navigation (Fig. 3a), robots navigate to their corresponding goal positions while actively adjusting trajectories (e.g., at $t = 1.7$s and $t = 3.4$s) to maintain safety margins and adhere to speed limits. In Balance (Fig. 3b), starting from a default configuration (Red-Orange-Yellow-Green-Blue), the robots transport the payload to its target position while coordinating their actions to keep a specified inter-robot spacing and to establish the specified spatial order (Orange-Red-Green-Yellow-Blue) by $t = 12.3$s. In Sampling (Fig. 3c), the robots collect samples from the high probability regions while forming a tight cluster with a specified radius (starting at $t = 1.9$s) and coordinating their motions to avoid the geofenced zone visualized by the red circle.

C. Results on High-Fidelity Multi-Robot Simulations

To intuitively demonstrate our CFPG approach, we perform a case study in a high-fidelity Unity simulation environment using Clearpath Husky robots running Robot Operating System 1 (ROS1). The robots are tasked with the Navigation scenario, where they must reach their assigned goals while avoiding collisions with other robots (no static obstacles are present). In addition to the nominal navigation objective, we also impose the same execution objectives (speed limit and safety margin) as constraints during inference. Additional experimental details are provided in the Supplementary Material. The qualitative results are demonstrated in Fig. 4. We can observe that the robots successfully navigate towards their goals while avoiding collisions with other robots. The robots actively adjust their paths in real-time to satisfy the safety constraints, resulting in smooth and efficient trajectories.

D. Validation on Physical Robot Teams

We further validate CFPG on a physical multi-robot system comprising AgileX LIMO robots running ROS2. The robots are similarly tasked with the Navigation objective in a real-world indoor setting without static obstacles, subject to the same execution objectives of speed limit and safety margin. We visualize the navigation scenario by creating a mixed-reality setup using an overhead projector mounted to the ceiling for real-time visualization, and use an OptiTrack motion capture system to retrieve robot pose information. Fig. 5 illustrates the execution of the policy. We observe the robots coordinating

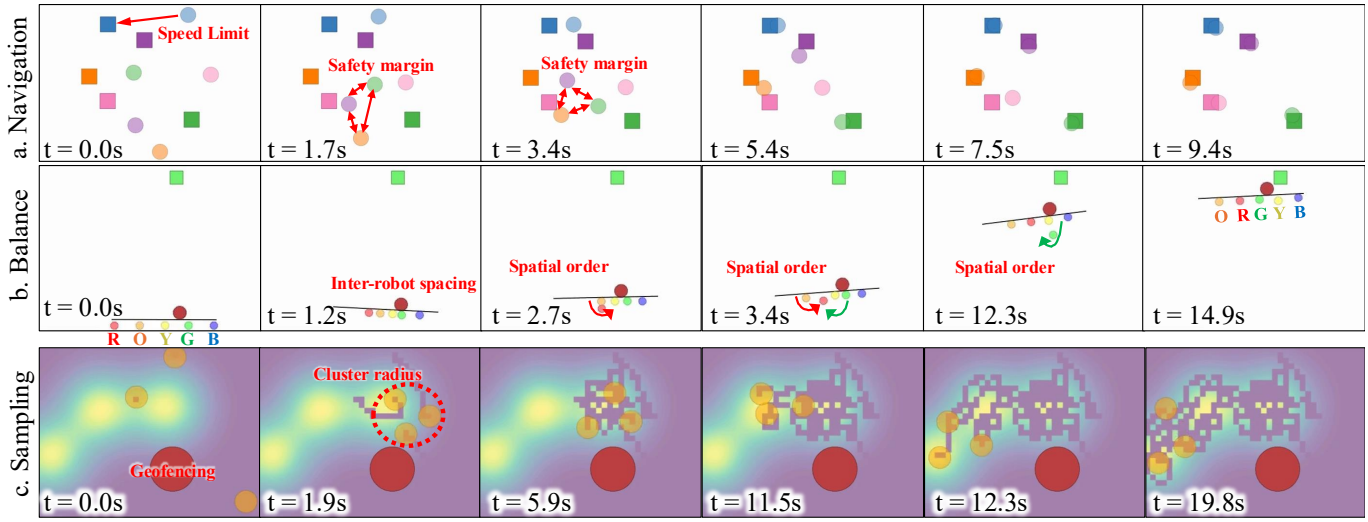

Fig. 3. The qualitative results demonstrate CFPG's ability to add new objectives during execution across the environments of (a) Navigation, where circles represent robots and the corresponding squares represent their goal locations; (b) Balance, where the smaller circles under the line represent robots, while the red circle is the payload that must be transported to the goal locations (green square); and (c) Sampling, where the orange circles represent robots, the larger red circle is the geofenced zone, and the background, ranging from blue to yellow, visualizes the relative spatial distribution of samples in the field, with yellow indicating higher density.

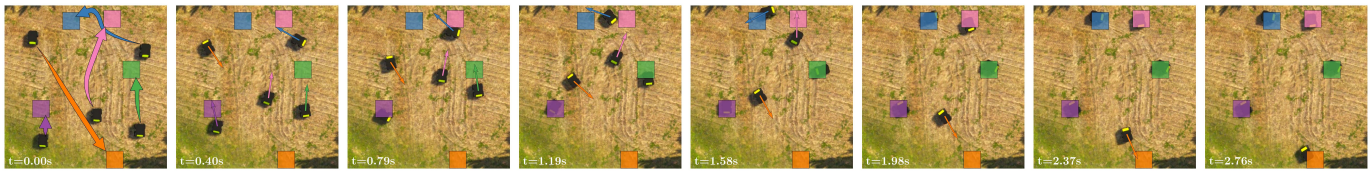

Fig. 4. Qualitative results from high-fidelity simulations rendered in Unity using Clearpath Husky robots running on ROS1.

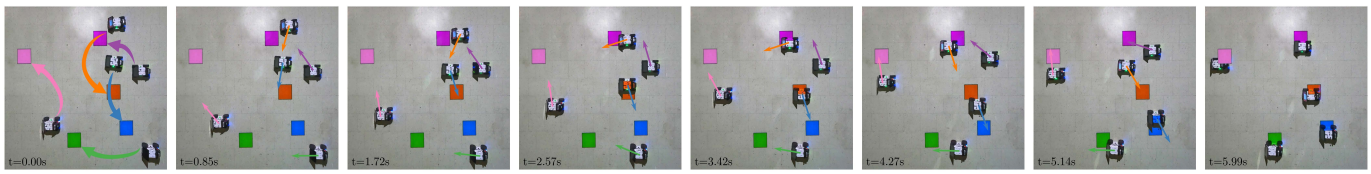

Fig. 5. Qualitative results from real-world experiments using LIMO robots operating fully autonomously under ROS 2 with Wi-Fi-based communication.

their motion to reach their respective targets without collision. CFPG effectively guides the robots to complete the task while satisfying the execution constraints, demonstrating its robustness and applicability to physical systems.

E. Discussion

Guidance Sensitivity. We analyzed the performance of CFPG with respect to the guidance strength η (Fig. 6). Zero guidance results in low success rates as the policy fails to satisfy new execution constraints. Conversely, excessively high guidance pulls the robots too far from the learned collaborative manifold, leading to reduced nominal task performance. This highlights the importance of selecting an appropriate guidance scale to balance both objectives.

Runtime Analysis. To evaluate computational feasibility, we analyzed the number of ODE solver steps T versus performance and latency (Fig. 7). CFPG achieves real-time execution ($>$ 10Hz) on embedded hardware (NVIDIA Jetson Orin Nano, CPU) with $T = 10$ steps, demonstrating that high-quality flow generation can be performed within strict latency budgets.

Scalability. The execution-time complexity of CFPG is $O(T \cdot (M \cdot K + M^2))$, where T is the number of ODE steps, M is the number of execution objectives, and $K \leq N$ is the number of communicating neighbors. This complexity scales linearly with T and the neighborhood size K, making it computationally scalable to larger multi-robot teams.

VI. Related Work

A. Multi-Objective Multi-Agent Reinforcement Learning

Multi-Agent Reinforcement Learning (MARL) extends reinforcement learning to systems where multiple agents must coordinate to achieve shared or individual goals. While standard MARL algorithms like MAPPO [9] and QMIX [28] optimize scalar rewards, Multi-Objective MARL addresses conflicting

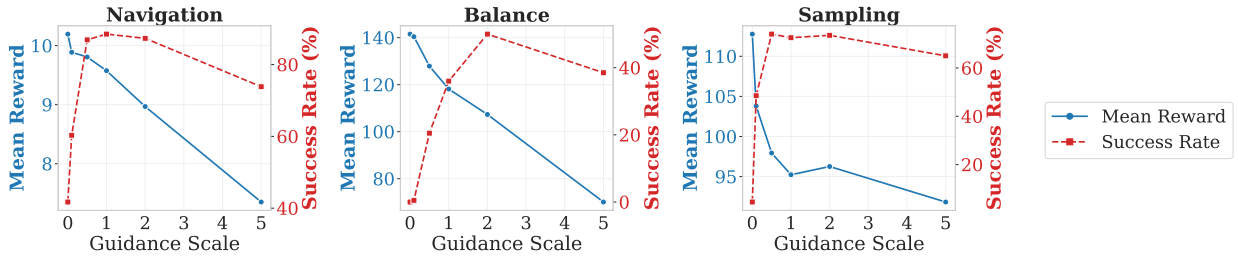

Fig. 6. Hyperparameter sensitivity analysis of the guidance scale (η). Moderate guidance balances execution success without degrading nominal performance.

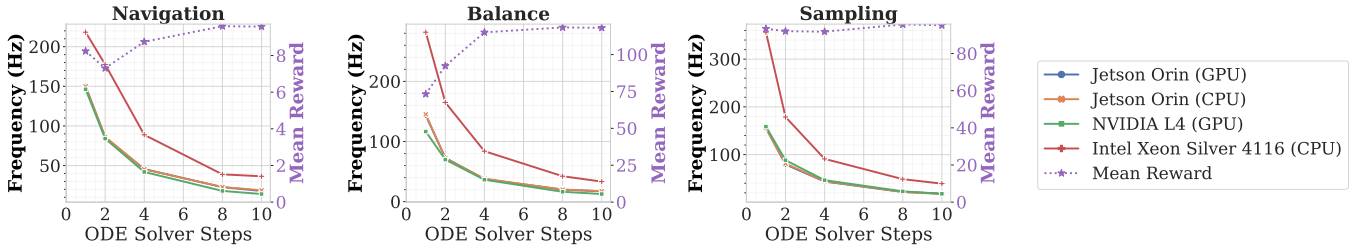

Fig. 7. Computational feasibility analysis in various environments illustrating the trade-off between control frequency and task performance as a function of the number of ODE solver steps.

goals using scalarization [10], [11] or Pareto approximation [29]. Specific approaches explore reward decomposition [30], social dilemmas [31], [32], and safety constraints [33], [34]. To mitigate optimization interference, gradient manipulation techniques like PCGrad [12], CAGrad [35], GradNorm [36], and MGDA [37] have been adapted to align multi-agent incentives [13] and ensure fairness [14]. However, these methods typically pre-define preferences into fixed policies during training, preventing adaptation to dynamic execution-time constraints. Our work overcomes this by decoupling collaboration learning from execution-time optimization using guided flow matching, with the added benefit of being able to learn multi-modal action distributions.

B. Diffusion Policy and Flow Matching

Generative models like diffusion [38]–[41] and flow matching [18], [19] have transformed robotics by capturing complex, multi-modal distributions. While diffusion models typically employ denoising score matching, diffusion policy [42]–[44] leverages them for imitation learning, surpassing reinforcement policy-based approaches. Flow matching shares these generative strengths but offers faster, simulation-free training and deterministic inference without complex noise schedulers. Recent works apply flow matching to control [15], [45]–[48] and utilize guidance [21], [22], [49]–[52] for steering outputs. However, existing methods are difficult to apply to multi-robot teams due to the reliance on offline datasets (which are difficult to collect especially for increasing number of robots) and lack conflict resolution for multiple objectives at execution time. Our approach bridges this by training multi-agent flow policies in a fully online and on-policy manner, and by introducing hierarchical gradient projection for execution-time conflict-aware guidance.

VII. Conclusion

We presented *Collaborative Flow Policy Guidance* (CFPG), a novel approach for learning multi-robot collaboration strategies that optimize a nominal objective and achieve adaptation to new execution objectives in a fully training-free manner. By introducing Multi-Agent Flow Policy Optimization (MAFPO), we showed that on-policy flow matching models can effectively capture the multimodal action distributions inherent in multi-robot collaboration. Furthermore, CFPG enables the direct optimization of previously unseen execution objectives during inference by leveraging the guidance property of flow matching, without requiring additional training. To balance new execution objectives with the nominal objective, we introduced a hierarchical gradient projection mechanism that resolves gradient conflicts during guidance. In addition, we provided theoretical analysis and proofs for CFPG and validated our approach empirically. Experimental results demonstrate that CFPG achieves superior performance and robustness across multiple multi-robot simulation environments, outperforming state-of-the-art baselines in adapting to dynamic execution-time requirements. CFPG currently assumes homogeneous agents and robots with full communication and pose observations in 2D environments. Future work includes extending CFPG to collaborative teams of heterogeneous robots, supporting varying communication topologies for real-world multi-robot deployment, and scaling to more complex tasks in unstructured 3D environments.

Acknowledgments

This work was partially supported by DARPA Young Faculty Award (YFA) D21AP10114-00, NSF CAREER Award IIS-2308492, DARPA Award N652362690001, DEVCOM ARL TBAM CRA W911NF2520024, and NSF Award IIS-2404386.

REFERENCES

[1] X. Cao, M. Li, Y. Tao, and P. Lu, "HMA-SAR: Multi-Agent Search and Rescue for Unknown Located Dynamic Targets in Completely Unknown Environments," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, pp. 5567–5574, 2024.

[2] J. P. Queralta, J. Taipalmaa, B. Can Pullinen, V. K. Sarker, T. Nguyen Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, and T. Westerlund, "Collaborative Multi-Robot Search and Rescue: Planning, Coordination, Perception, and Active Vision," *IEEE Access*, vol. 8, pp. 191 617–191 643, 2020.

[3] Z. He, X. Zhang, S. Jones, S. Hauert, D. Zhang, and N. F. Lepora, "TacMMs: Tactile Mobile Manipulators for Warehouse Automation," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4729–4736, 2023.

[4] W. J. Jose and H. Zhang, "Learning for Dynamic Subteaming and Voluntary Waiting in Heterogeneous Multi-Robot Collaborative Scheduling," in *IEEE International Conference on Robotics and Automation*, 2024.

[5] M. J. Schuster, M. G. Müller, S. G. Brunner, H. Lehner, P. Lehner, R. Sakagami, A. Dömel, L. Meyer, B. Vodermayer, R. Giubilato, M. Vayugundla, J. Reill, F. Steidle, I. von Bargen, K. Bussmann, R. Belder, P. Lutz, W. Stürzl, M. Smíšek, M. Maier, S. Stoneman, A. F. Prince, B. Rebele, M. Durner, E. Staudinger, S. Zhang, R. Pöhlmann, E. Bischoff, C. Braun, S. Schröder, E. Dietz, S. Frohmann, A. Börner, H.-W. Hübers, B. Foing, R. Triebel, A. O. Albu-Schäffer, and A. Wedler, "The ARCHES Space-Analogue Demonstration Mission: Towards Heterogeneous Teams of Autonomous Robots for Collaborative Scientific Sampling in Planetary Exploration," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5315–5322, 2020.

[6] H. Farivarnejad, A. S. Lafmejani, and S. Berman, "Fully Decentralized Controller for Multi-Robot Collective Transport in Space Applications," in *IEEE Aerospace Conference*, 2021.

[7] K. R. Jensen-Nau, T. Hermans, and K. K. Leang, "Near-Optimal Area-Coverage Path Planning of Energy-Constrained Aerial Robots With Application in Autonomous Environmental Monitoring," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 3, pp. 1453–1468, 2021.

[8] X. He, J. R. Bourne, J. A. Steiner, C. Mortensen, K. C. Hoffman, C. J. Dudley, B. Rogers, D. M. Cropek, and K. K. Leang, "Autonomous Chemical-Sensing Aerial Robot for Urban/Suburban Environmental Monitoring," *IEEE Systems Journal*, vol. 13, no. 3, pp. 3524–3535, 2019.

[9] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games," in *Neural Information Processing Systems*, 2022.

[10] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A Survey of Multi-Objective Sequential Decision-Making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[11] A. Abels, D. Roijers, T. Lenaerts, A. Nowé, and D. Steckelmacher, "Dynamic Weights in Multi-Objective Deep Reinforcement Learning," in *International Conference on Machine Learning*, 2019.

[12] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient Surgery for Multi-Task Learning," in *Neural Information Processing Systems*, 2020.

[13] Y. Li, W. Zhang, J. Wang, and S. Zhang, "Aligning Individual and Collective Objectives in Multi-Agent Cooperation," in *Neural Information Processing Systems*, 2024.

[14] W. Kim and K. Sycara, "Fair Cooperation in Mixed-Motive Games via Conflict-Aware Gradient Adjustment," in *Neural Information Processing Systems*, 2025.

[15] D. McAllister, S. Ge, B. Yi, C. M. Kim, E. Weber, H. Choi, H. Feng, and A. Kanazawa, "Flow Matching Policy Gradients," in *International Conference on Learning Representations*, 2026.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[17] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-Dimensional Continuous Control Using Generalized Advantage Estimation," in *International Conference on Learning Representations*, 2016.

[18] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow Matching for Generative Modeling," in *International Conference on Learning Representations*, 2023.

[19] Y. Lipman, M. Havasi, P. Holderrieth, N. Shaul, M. Le, B. Karrer, R. T. Q. Chen, D. Lopez-Paz, H. Ben-Hamu, and I. Gat, "Flow Matching Guide and Code," *arXiv preprint arXiv:2412.06264*, 2024.

[20] Z. Xie, Q. Zhang, F. Yang, M. Hutter, and R. Xu, "Simple Policy Optimization," in *International Conference on Machine Learning*, 2025.

[21] Y. Guo, H. Yuan, Y. Yang, M. Chen, and M. Wang, "Gradient Guidance for Diffusion Models: An Optimization Perspective," in *Neural Information Processing Systems*, 2024.

[22] R. Feng, C. Yu, W. Deng, P. Hu, and T. Wu, "On the Guidance of Flow Matching," in *International Conference on Machine Learning*, 2025.

[23] Y. Nesterov, *Lectures on Convex Optimization*, ser. Springer Optimization and Its Applications. Cham: Springer International Publishing, 2018, vol. 137.

[24] T.-Y. Yang, J. Rosca, K. Narasimhan, and P. J. Ramadge, "Projection-Based Constrained Policy Optimization," in *International Conference on Learning Representations*, 2020.

[25] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe Reinforcement Learning via Shielding," in *AAAI Conference on Artificial Intelligence*, 2018.

[26] I. ElSayed-Aly, S. Bharadwaj, C. Amato, R. Ehlers, U. Topcu, and L. Feng, "Safe Multi-Agent Reinforcement Learning via Shielding," in *International Conference on Autonomous Agents and Multiagent Systems*, 2021.

[27] M. Bettini, R. Kortvelesy, J. Blumenkamp, and A. Prorok, "VMAS: A Vectorized Multi-Agent Simulator for Collective Robot Learning," in *International Symposium on Distributed Autonomous Robotic Systems*, 2022.

[28] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning," in *International Conference on Machine Learning*, 2018.

[29] K. V. Moffaert and A. Nowé, "Multi-Objective Reinforcement Learning using Sets of Pareto Dominating Policies," *Journal of Machine Learning Research*, vol. 15, no. 107, pp. 3663–3692, 2014.

[30] R. Yang, X. Sun, and K. Narasimhan, "A Generalized Algorithm for Multi-Objective Reinforcement Learning and Policy Adaptation," in *Neural Information Processing Systems*, 2019.

[31] J. Z. Leibo, V. Zambaldi, and M. Lanctot, "Multi-agent Reinforcement Learning in Sequential Social Dilemmas," in *International Conference on Autonomous Agents and Multiagent Systems*, 2017.

[32] E. Hughes, J. Z. Leibo, M. Phillips, K. Tuyls, E. Dueñez-Guzman, A. García Castañeda, I. Dunning, T. Zhu, K. McKee, R. Koster, H. Roff, and T. Graepel, "Inequity aversion improves cooperation in intertemporal social dilemmas," in *Neural Information Processing Systems*, 2018.

[33] S. Zhang, O. So, M. Black, Z. Serlin, and C. Fan, "Solving Multi-Agent Safe Optimal Control with Distributed Epigraph Form MARL," in *Robotics: Science and Systems*, 2025.

[34] S. Gu, L. Shi, Y. Ding, A. Knoll, C. Spanos, A. Wierman, and M. Jin, "Enhancing Efficiency of Safe Reinforcement Learning via Sample Manipulation," in *Neural Information Processing Systems*, 2024.

[35] B. Liu, X. Liu, X. Jin, P. Stone, and Q. Liu, "Conflict-Averse Gradient Descent for Multi-task learning," in *Neural Information Processing Systems*, 2021.

[36] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks," in *International Conference on Machine Learning*, 2018.

[37] J.-A. Désidéri, "Multiple-gradient descent algorithm (MGDA) for multi-objective optimization," *Comptes Rendus. Mathématique*, vol. 350, no. 5-6, pp. 313–318, 2012.

[38] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," in *Neural Information Processing Systems*, 2020.

[39] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-Based Generative Modeling Through Stochastic Differential Equations," in *International Conference on Learning Representations*, 2021.

[40] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," in *International Conference on Learning Representations*, 2021.

[41] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M.-H. Yang, "Diffusion Models: A Comprehensive Survey of Methods and Applications," *ACM Computing Surveys*, vol. 56, no. 4, pp. 1–39, 2023.

[42] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Visuomotor Policy Learning via Action Diffusion," in *Robotics: Science and Systems*, 2023.

[43] E. Ng, Z. Liu, and M. Kennedy, "Diffusion Co-Policy for Synergistic Human-Robot Collaborative Tasks," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 215–222, 2024.

[44] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration," in *IEEE International Conference on Robotics and Automation*, 2024.
[45] M. M. Sun, A. Pinosky, and T. Murphey, "Flow Matching Ergodic Coverage," in *Robotics: Science and Systems*, 2025.
[46] A. Murillo-Gonzalez and L. Liu, "Action Flow Matching for Continual Robot Learning," in *Robotics: Science and Systems*, 2025.
[47] Y. Yuan, C. Pal, and X. Liu, "ParetoFlow: Guided Flows in Multi-Objective Optimization," in *International Conference on Learning Representations*, 2025.
[48] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky, "π_0: A Vision-Language-Action Flow Model for General Robot Control," in *Robotics: Science and Systems*, 2025.
[49] P. Dhariwal and A. Nichol, "Diffusion Models Beat GANs on Image Synthesis," in *Neural Information Processing Systems*, 2021.
[50] J. Ho and T. Salimans, "Classifier-Free Diffusion Guidance," in *Neural Information Processing Systems Workshop on Deep Generative Models and Downstream Applications*, 2021.
[51] L. Wang, C. Cheng, Y. Liao, and Y. Qu, "Training Free Guided Flow Matching with Optimal Control," in *International Conference on Learning Representations*, 2025.
[52] A. Bansal, H.-M. Chu, A. Schwarzschild, S. Sengupta, M. Goldblum, J. Geiping, and T. Goldstein, "Universal Guidance for Diffusion Models," in *International Conference on Learning Representations*, 2024.