CONFIDENTIAL. Limited circulation. For review only.
IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.
Cite as T-ASE paper.

Risk-Aware and Scalable Hierarchical Motion Planning for Large-Scale Robotic Swarms via CVaR-Constrained MPC

Xuru Yang, Yuqiao Zhao, Yunze Hu, Zongru Yang, Pingping Zhu, *Member, IEEE*
Ying Sun, *Member, IEEE* and Chang Liu, *Member, IEEE*

***Abstract*— Motion planning for large-scale robotic swarms presents significant challenges in terms of scalability and safety assurance in cluttered environments. To address these issues, this manuscript proposes a Closed-loop hierarchical Risk-aware swarm mOtion planner using Conditional ValuE at Risk (C-ROVER) that enables safe and efficient navigation for swarm robotic systems. The hierarchical structure of C-ROVER comprises a macroscopic planning stage that models the swarm state with Gaussian Mixture Models (GMMs) and generates trajectories for the swarm GMM, followed by a microscopic control stage that computes individual robot control using distributed model predictive control to track the GMM trajectories while achieving robot-level collision avoidance. Robot positions are periodically used to update the swarm GMM, closing the hierarchical planning and control loop. To achieve collision risk-awareness between the swarm and environmental obstacles at the macroscopic stage, C-ROVER leverages the stochastic Signed Distance Function to characterize the distance between the swarm GMM and obstacles, which is proven to follow a GMM. Then C-ROVER proposes an analytical expression of Conditional Value-at-Risk (CVaR) of a GMM to enable the swarm collision risk mitigation. Furthermore, C-ROVER designs a novel risk-aware space discretization approach to enhance the ability to navigate constrained spaces. To achieve efficient online motion planning, C-ROVER develops a convergent sequential convex programming approach for macroscopic planning, leveraging the concavity of CVaR constraints. C-ROVER has been evaluated through various simulations and real-world experiments, demonstrating its capability to ensure safe, scalable, and real-time swarm navigation in cluttered scenarios.**

I. Introduction

Large-scale swarm robotic systems comprised of hundreds or thousands of autonomous and cooperative robots are attracting burgeoning research interest due to their superior efficiency and robustness in applications such as target detection [1], cooperative object transport [2], and search and rescue [3]. Despite considerable attention from fields such as swarm robotics [4, 5], optimal control [6–8], and multi-agent reinforcement learning [9, 10], motion planning for large-scale swarm robotic systems remains challenging, particularly in terms of scalability and safety maintenance.

Scaling up robotic swarms significantly increases the complexity of motion planning and control. Traditional *microscopic approaches* [5, 11–13], which directly compute solutions for each individual robot, provide granular control over each robot's behavior but are computationally intractable as

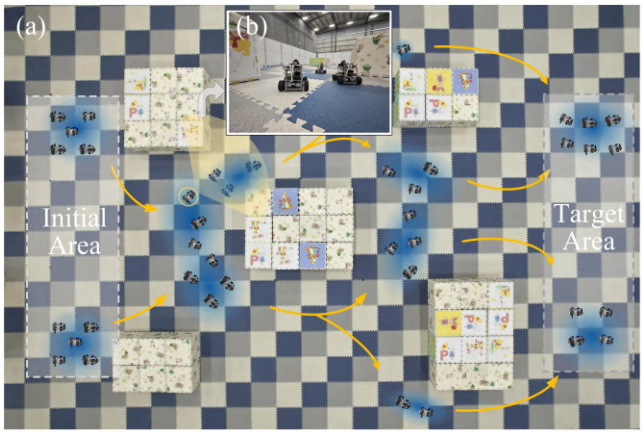

Fig. 1. Robot swarm motion planning using C-ROVER in a cluttered workspace. (a) The robots set off from the initial area and navigate toward the target area. The orange paths trace the swarm's transport trajectories, while the blue shaded areas depict snapshots of GMMs at several moments, representing the robots' distribution. (b) The first-person viewpoint of the robot is highlighted by a yellow circle in (a), with the yellow sector indicating the field of view.

TABLE I
Summary of Common Notation

Symbol	Meaning
$\mathcal{N}$	Normal distribution
$\wp$	Probability distribution of the robot swarm represented as a GMM
ω_j	Weight of the jth Gaussian component in the GMM
μ_j, Σ_j	Mean and Covariance matrix of the jth Gaussian component
$g_j = \mathcal{N}(\mu_j, \Sigma_j)$	jth Gaussian component
$\pi_k(i, j)$	Transport policy from Gaussian component i at step k to j at $k+1$
$W_2(\cdot, \cdot)$	Wasserstein distance between two distributions
$sd(X, O_\iota)$	Signed distance from position X to obstacle O_ι
$\text{CVaR}_\alpha(Y)$	Conditional Value-at-Risk of random variable Y at risk tolerance level α
N_o, N_c, N_r	Number of obstacles, Gaussian components, and robots
$\underline{N}$	The set $\{1, 2, \ldots, N\}$ for any $N \in \mathbb{N}_+$
h	MPC planning horizon
u_i^k	Control input of robot i at time step k
x_i^k	Position of robot i at time step k
$f_{I_k, I_{k+1}, \ldots, I_{\text{targ}}}$	GMM-weight flow variable along the spatio-temporal path $I_k, I_{k+1}, \ldots, I_{\text{targ}}$

This work is sponsored by Beijing Nova Program (20220484056, 20240484498) and National Natural Science Foundation of China (62203018) (Corresponding author: Chang Liu).

Xuru Yang, Yuqiao Zhao, Yunze Hu, Zongru Yang, and Chang Liu are with School of Advanced Manufacturing and Robotics, Peking University, Beijing 100871, China (emails: xuru.yang@outlook.com; yuqiaozhao@stu.pku.edu.cn; hu_yun_ze@stu.pku.edu.cn; yzr@stu.pku.edu.cn; changliucoe@pku.edu.cn).

Pingping Zhu is with the Department of Biomedical and Electrical Engineering and Department of Computer Science at Marshall University, Huntington, WV, USA (emails: zhup@marshall.edu).

Ying Sun is with School of Electrical Engineering and Computer Science, Pennsylvania State University, State College, PA, USA (emails: ybs5190@psu.edu).

Xuru Yang and Yuqiao Zhao contributed equally to this work.

Manuscript 6295 submitted to 2026 IEEE International Conference on Robotics and Automation (ICRA). Received December 21, 2025.

swarm size grows. While many of these approaches incorporate distributed or parallel computing techniques to mitigate computational burden, their applications remain limited to swarms of moderate scale. *Macroscopic approaches* that adopt a collective perspective of the entire swarm [8, 14–18] represent an effective attempt to reduce computational complexity and thus greatly enhance scalability. However, such methods often excessively simplify the kinematic and dynamic models of the robots and fail to account for collision avoidance with environmental obstacles or among robots, which limits their direct applicability to real-world swarm deployments.

Hierarchical approaches that integrate the advantages of both macroscopic and microscopic methods have effectively reached a balance between computational cost and collision avoidance. Recent works [4, 19–21] propose to provide the swarm with rough reference trajectories at the upper level, followed by individual motion controllers to track the references at the lower level. However, the reference trajectories are typically constrained within a fixed formation or a group, limiting the maneuver flexibility of the swarm.

To address these gaps, this work proposes a Closed-loop hierarchical Risk-aware swarm mOtion planner using conditional ValuE at Risk (C-ROVER) that facilitates safe and efficient navigation for swarm robotic systems in cluttered environments (Fig. 1). C-ROVER decomposes the motion planning problem into a *macroscopic planning stage* and a *microscopic control stage*. The macroscopic planning stage represents the swarm state by Gaussian Mixture Models (GMMs) and generates trajectories of the swarm GMM with a Model Predictive Control (MPC) scheme that incorporates Conditional Value-at-Risk (CVaR) to strengthen collision avoidance. The resulting GMM trajectories guide the microscopic stage, where individual robots perform trajectory tracking and collision avoidance using distributed MPC (DMPC). Robot positions are periodically utilized to update the GMM so that the swarm GMM can accurately reflect the distribution of robots, thereby closing the loop of hierarchical planning and control. The specific contributions of this work can be summarized as follows.

- We develop a stochastic Signed Distance Function (SDF) formulation within the macroscopic planning framework to model spatial proximity between the swarm and obstacles. We prove that the stochastic SDF follows a GMM distribution, which enables the derivation of an analytical CVaR expression for quantifying swarm collision risk.
- We propose a risk-aware discretization approach for the macroscopic planning space, represented by a set of Gaussian distributions. A convex optimization problem is formulated to optimize the covariance matrices with respect to CVaR-based collision risk.
- We formulate a non-convex MPC problem with CVaR constraints for GMM-based trajectory planning. To enable efficient online computation, we develop a provably convergent sequential programming algorithm that constructs a conservative surrogate problem by leveraging the concavity property of CVaR.
- We validate C-ROVER through extensive simulations and real-world experiments, demonstrating its high scalability, safety, and real-time performance for swarm navigation in cluttered environments.

II. Related Work

This section discusses previous research in motion planning for large-scale robotic systems regarding the two dominant challenges: scalability enhancement and collision avoidance.

A. Scalability Enhancement

Scalability is one of the most prominent challenges for large-scale swarm motion planning. One straightforward approach, called microscopic approach, conducts motion planning at the individual level [5, 11–13]. However, microscopic approaches become increasingly computationally expensive as swarm size grows. While distributed computing partially alleviates the computational cost [5, 12], it often struggles to maintain real-time performance and coordination efficiency in large-scale swarms.

Another category, known as the macroscopic approach, models the swarm as a collective system and enables scalable motion planning within macroscopic state space that is independent of the swarm size. A prevalent concept is density control [8, 14–16], which involves representing the swarm using Probability Density Functions (PDFs) and formulating optimal control problems over the PDF space to reshape the distribution towards a desired one. Another prominent category of macroscopic approaches relies on mean-field theory [17, 18], which assumes that each robot's behavior is influenced by the average behavior of the entire group. These studies commonly rely on the assumptions of obstacle-free environments and ignore robot dynamics constraints, which makes macroscopic approaches theoretically convenient but with limited real-world applicability.

Recently, hierarchical approaches have garnered heightened attention. The hierarchical approach produces reference trajectories for the entire swarm at an upper level, and generates the robots' control input at the lower level to track the reference trajectories. Alonso et al. [19] leverage formation parameters to represent the swarm and compute collision-free formation transformation at the upper level. Mao et al. [4] design an optimal virtual tube for swarm robotics trajectory planning, based on which each robot is assigned an optimal reference trajectory. The major drawback of [4, 19] is constraining all robots in a group and not allowing the swarm to split. Adaptive online Distributed Optimal Control (ADOC) proposed by Zhu et al. [6] overcomes this limitation by representing the macroscopic state of the swarm with GMMs and optimizing GMM trajectories at the upper level. The primary limitation of ADOC lies in its greedy planning framework and the use of a penalty-based GMM collision avoidance scheme, which may lead to risky trajectories in cluttered environments. Yang et al. [20] propose a hierarchical non-myopic motion planning

approach for swarm robots. However, the upper level planner lacks convergence guarantee and has limited collision risk awareness, making it difficult to navigate constrained spaces. Besides, the lower level controller assumes a point-mass kinematic model of robots, hindering the applications in real-world systems. Moreover, the method operates in an open-loop manner, where the upper level planning ignores the actual robots' states, leading to issues where robots may not be able to track the reference trajectories from the upper level planner and thus become outliers.

B. Collision Avoidance

One category of collision avoidance approaches employs deterministic collision avoidance methodology, which assumes that robots perceive the states of their own and obstacles precisely. One typical method, Optimal Reciprocal Collision Avoidance (ORCA) [22], implements distributed collision avoidance through velocity obstacles for homogeneous omnidirectional robots. The Separating Hyperplane method [23] approximates obstacles via hyperplanes to simplify optimization computations. Similar approaches include constructing safe corridors [24], collision-free convex polytopes [19], and obstacle-free virtual tubes [4]. These methods guide swarm motion by defining obstacle-free regions but suffer from over-conservative trajectory planning due to expanded infeasible regions.

Another category models the environment or robot state using probabilistic representations and adopts probabilistic collision avoidance methodologies. For example, ADOC [6] introduces a penalty term in the objective function proportional to the inner product between the swarm's PDF and the environment's occupancy map. However, in cluttered environments, this soft constraint may result in trajectories that pass near obstacles, thereby increasing the computational effort required at the agent level to generate collision-free controls. A prevalent approach incorporates probabilistic risk measures. Chance-constrained methods [25–27] are widely used in risk-aware motion planning, enforcing probabilistic safety constraints below predefined risk thresholds. However, chance constraints do not account for the collision severity, and the tail of the planned state distribution may intrude deeply into obstacles, resulting in a high likelihood of collisions for tail robots. Value-at-Risk estimates the maximum loss at a given confidence level but still ignores losses beyond that point. By contrast, CVaR [28] penalizes expected tail loss, so safety margins are allocated where severe outcomes are likely while low-risk regions are not uniformly tightened. Although both frameworks can trade risk for efficiency, CVaR provides a more targeted allocation of caution and achieves a more balanced safety and efficiency tradeoff.

While CVaR has been applied to single-robot [29, 30] and small-scale multi-robot systems [13], current implementations primarily handle Gaussian risk distributions with a closed-form CVaR expression. For non-Gaussian distributions, extensive computational resources are required for CVaR estimation. A critical research gap remains in developing computationally efficient CVaR-based risk measures for non-Gaussian distributions to apply CVaR in large-scale swarm collision avoidance.

III. Preliminary and Problem Formulation

A. Wasserstein Metric

The Wasserstein metric measures the distance between two probability distributions on a given metric space, and is widely used in optimal transport theories [31]. We first consider two Gaussian PDFs denoted by $g_1 = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) \in \mathcal{P}(\mathbb{R}^d)$ and $g_2 = \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2) \in \mathcal{P}(\mathbb{R}^d)$ with $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ denoting the Gaussian means, $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ denoting the Gaussian covariances, and $\mathcal{P}(\Omega)$ denoting the set of all the probability distributions defined on the sample space Ω. The Wasserstein metric W_2 between g_1 and g_2 can be analytically calculated by [31]

$$W_2(g_1, g_2) = \Big\{ \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|^2 + tr\left[\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 - 2\left(\boldsymbol{\Sigma}_1^{1/2}\boldsymbol{\Sigma}_2\boldsymbol{\Sigma}_1^{1/2}\right)^{1/2}\right] \Big\}^{1/2}, \tag{1}$$

where $tr(\cdot)$ indicates the trace operator, and $\|\cdot\|$ denotes the Euclidean norm. The Wasserstein metric between two GMMs can be approximated utilizing eq. (1) [31]. Specifically, consider two GMMs with N_1 and N_2 Gaussian components (GCs) denoted by $\wp_1 = \sum_{i=1}^{N_1} \omega_1^i g_1^i, \wp_2 = \sum_{j=1}^{N_2} \omega_2^j g_2^j$, where g_1^i, g_2^j denote the ith GC and the jth GCs of $\wp_1$ and $\wp_2$, respectively, and ω_1^i, ω_2^j denote their corresponding weights that satisfy $\sum_{i=1}^{N_1} \omega_1^i = 1$ and $\sum_{j=1}^{N_2} \omega_2^j = 1$. The Wasserstein metric $W_2(\wp_1, \wp_2)$ can be approximated as:

$$W_2(\wp_1, \wp_2) \triangleq \left\{ \min_{\pi \in \Pi(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)} \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} [W_2(g_1^i, g_2^j)]^2 \pi(i,j) \right\}^{1/2}, \tag{2}$$

where $\boldsymbol{\omega}_1 = [\omega_1^1, \cdots, \omega_1^{N_1}]^T$ and $\boldsymbol{\omega}_2 = [\omega_2^1, \cdots, \omega_2^{N_2}]^T$ denote the weight vector, and $\Pi(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$ denotes the space of joint distributions between $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$. From an intuitive perspective, $\pi(i,j)$ represents the weight transported from the ith GC in $\wp_1$ to the jth GC in $\wp_2$.

B. Signed Distance Function

The SDF measures the orthogonal distance between a point and the boundary of a set in a metric space with the sign indicating whether or not the point is in the interior of the set. In particular, the SDF between a point $\boldsymbol{p} \in \mathbb{R}^2$ and an obstacle $\mathcal{O}_\iota \subset \mathbb{R}^2$ can be calculated as follows,

$$sd(\boldsymbol{p}, \mathcal{O}_\iota) = \begin{cases} -d(\boldsymbol{p}, \partial\mathcal{O}_\iota), & \text{if } \boldsymbol{p} \in \mathcal{O}_\iota \\ d(\boldsymbol{p}, \partial\mathcal{O}_\iota), & \text{if } \boldsymbol{p} \notin \mathcal{O}_\iota \end{cases}, \tag{3}$$

where $d(\boldsymbol{p}, \partial\mathcal{O}_\iota)$ is the minimum distance from $\boldsymbol{p}$ to the boundary of $\mathcal{O}_\iota$, noted as $\partial\mathcal{O}_\iota$. The closest point to $\boldsymbol{p}$ on $\partial\mathcal{O}_\iota$ is denoted as $\boldsymbol{o}_\iota^*$, and the normal vector along the SDF direction can be calculated as follows,

$$\boldsymbol{n} = sgn(sd(\boldsymbol{p}, \mathcal{O}_\iota)) \cdot (\boldsymbol{p} - \boldsymbol{o}_\iota^*)/\|\boldsymbol{p} - \boldsymbol{o}_\iota^*\|, \tag{4}$$

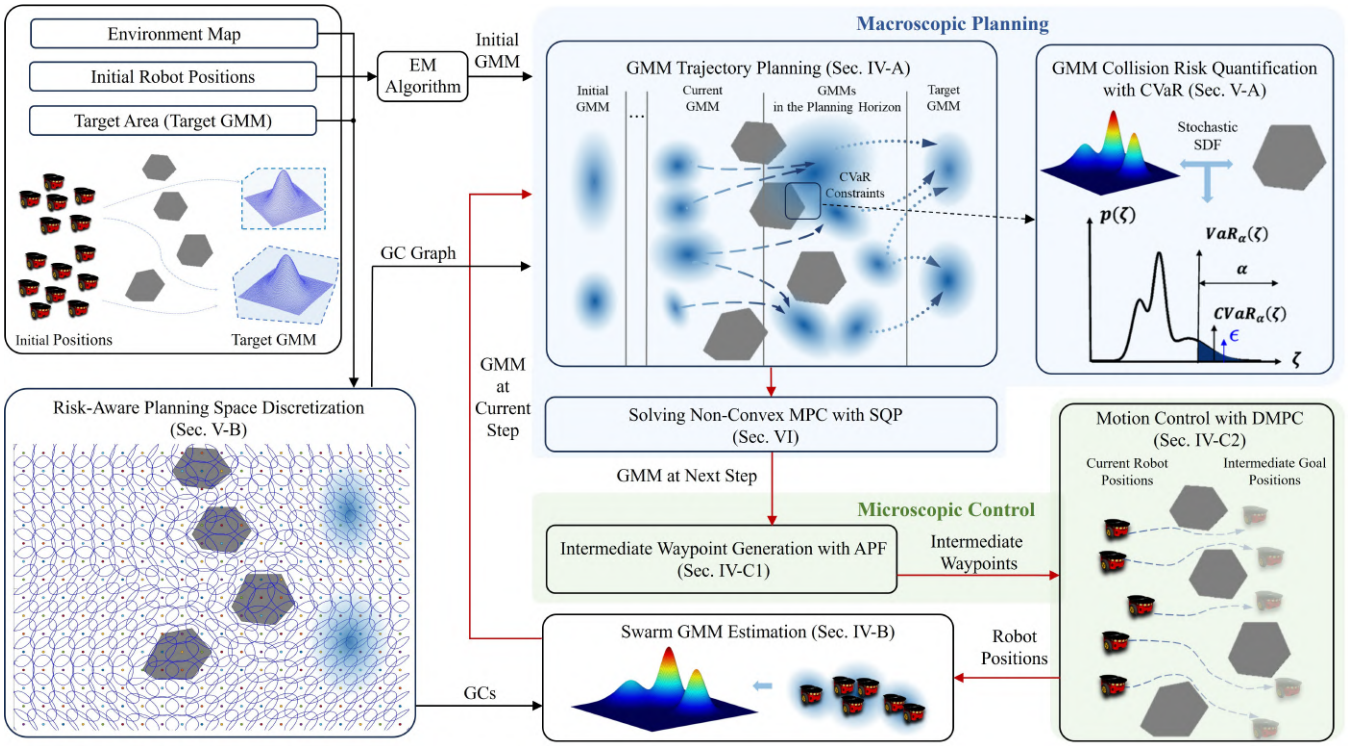

Fig. 2. Overview of C-ROVER. The blue and green shaded backgrounds indicate the macroscopic planning module and the microscopic control module, respectively, with the red arrows indicating the closed-loop process. Ellipses with blue contours and colored centers in the bottom left area of the figure denote GCs for planning space discretization. The blue density clouds denote GMMs.

where $sgn(\cdot)$ is the sign function. The SDF and its associated normal vector $\boldsymbol{n}$ can be efficiently calculated utilizing Gilbert–Johnson–Keerthi (GJK) algorithm [32] when the point is exterior to the set and Expanding Polytope Algorithm (EPA) [33] otherwise.

C. Value-at-Risk and Conditional Value-at-Risk

The VaR represents the minimum possible value of risk that can be achieved given a risk tolerance level $\alpha \in (0, 1]$. Specifically, the VaR of a random variable (r.v.) ζ under α is defined as

$$VaR_\alpha(\zeta) = \min\{z | Pr(\zeta \leqslant z) \geqslant 1 - \alpha\}, \tag{5}$$

where $Pr(\cdot)$ denotes the probability. The CVaR of ζ under α represents the expectation of ζ given the condition that ζ exceeds $VaR_\alpha(\zeta)$, i.e., $CVaR_\alpha(\zeta) = \mathbb{E}[\zeta | \zeta \geqslant VaR_\alpha(\zeta)]$ with $\mathbb{E}$ denoting the expectation operator. For ζ with a continuous distribution, $CVaR_\alpha(\zeta)$ can be calculated by [34]

$$CVaR_\alpha(\zeta) = \min_{z \in \mathbb{R}} \mathbb{E}[z + (\zeta - z)^+ / \alpha], \tag{6}$$

where the function $(\cdot)^+ = \max(\cdot, 0)$.

Let $\phi(\cdot)$ and $\Phi(\cdot)$ denote the PDF and the cumulative distribution function (CDF) of a standard normal distribution, respectively. The CVaR of a Gaussian r.v. $\zeta \sim \mathcal{N}(\mu, \sigma^2)$ has the following closed-form expression [35],

$$CVaR_\alpha(\zeta) = \mu + \sigma\phi(\Phi^{-1}(1 - \alpha))/\alpha. \tag{7}$$

D. Problem Formulation

To simplify notation, we define a set notation $\underline{N} = \{1, 2, \cdots, N\}$ for any $N \in \mathbb{N}_+$. Consider a workspace $\mathcal{W} \subset \mathbb{R}^2$ that contains N_o static obstacles $\mathcal{O}_\iota \subset \mathcal{W}, \iota \in \underline{N_o}$, and let $\mathcal{O} = \bigcup_{\iota=1}^{N_o} \mathcal{O}_\iota$ represent the union of all obstacles. For simplicity, $\mathcal{O}_\iota$ is assumed to be convex[1].

We consider a robot swarm with N_r robots with $\boldsymbol{x}_i(k)$ and $\boldsymbol{u}_i(k)$ denoting the state and control input of the ith robot at the discrete time step k, respectively. The objective of the proposed swarm motion planning problem is to safely transport robots from an initial zone to a designated target area with the shortest trajectory in obstacle-cluttered environments.

We provide a collective perspective to view this above-mentioned swarm transport task. We denote the robot swarm state with a time-varying r.v. $\boldsymbol{X}(k)$ following a distribution $\wp_k$, where the position of each robot $\boldsymbol{x}_i(k)$ can be considered as a sample from $\boldsymbol{X}(k)$. Due to the universal approximation property of GMM [36], we choose to use a GMM, denoted by $\wp_k = \sum_{j=1}^{N_k} \omega_k^j g_k^j$ with $\sum_{j=1}^{N_k} \omega_k^j = 1$, to represent the state of an arbitrary robot swarm. Here N_k denotes the number of GCs in the GMM and $g_k^j = \mathcal{N}(\boldsymbol{\mu}_k^j, \boldsymbol{\Sigma}_k^j)$ is the jth Gaussian distribution with mean $\boldsymbol{\mu}_k^j$, covariance matrix $\boldsymbol{\Sigma}_k^j$, and weight $\omega_k^j \geqslant 0$. From a collective standpoint, the primary objective of swarm transport is to generate the transformation of the swarm GMM from an initial distribution $\wp_0 \in \mathcal{G}(\mathcal{W})$

[1]For non-convex obstacles, the convex hull or convex decomposition can be applied to obtain convex obstacles.

to a desired target distribution $\wp_{\text{targ}} \in \mathcal{G}(\mathcal{W})$, where $\mathcal{G}(\mathcal{W})$ denotes the GMM distribution space.

IV. Closed-Loop Hierarchical Framework

We propose a closed-loop hierarchical framework, C-ROVER, to address the swarm motion planning problem by incorporating both the individual and collective perspectives (Fig. 2). The system takes as input the environment map, initial robot positions, and the target area represented as a GMM. The proposed framework comprises a macroscopic planning stage focusing on trajectory planning of the swarm GMM and a microscopic control stage for motion control of individual robots. The GMM planning space is discretized into a set of collision risk-aware GCs (Sec. V-B). At each planning step, a GMM representation of the swarm is constructed based on the spatial distribution of individual robot positions (Sec. IV-B). The macroscopic planning stage then generates the risk-aware GMM trajectory by solving a MPC (Sec. IV-A) with CVaR constraints (Sec. V-A) using a sequential optimization algorithm (Sec. VI). We will show that the computational complexity of macroscopic planning only depends on GMM and is independent of the swarm size, leading to highly efficient macroscopic planning. The planned GMM trajectories then serve as references for the microscopic control, where C-ROVER first generates intermediate waypoints for each robot with an artificial potential field (APF) approach to keep track of the swarm GMM and avoids collisions with obstacles and neighboring robots (Sec. IV-C.1). A DMPC is then designed to navigate robots to track their waypoints while satisfying the robots' kinematics constraints and ensuring collision avoidance (Sec. IV-C.2). Robot positions are periodically used to re-estimate the swarm GMM, which is used for the next GMM planning cycle, thus closing the planning and control loop.

A. Macroscopic Planning Stage

The macroscopic planning stage is structured as a swarm GMM optimal transport problem (P0) that aims to obtain the GMM transport strategy from an initial distribution $\wp_0 \in \mathcal{G}(\mathcal{W})$ to a target distribution $\wp_{targ} \in \mathcal{G}(\mathcal{W})$ while minimizing the overall transport cost and adhering to collision risk constraints, defined as follows,

$$\min_{\wp_1,\wp_2,\ldots,\wp_{T_f}} J \triangleq \sum_{k=1}^{T_f} \lambda_k W_2^2(\wp_{k-1}, \wp_k) \tag{8a}$$

$$s.t. \quad CVaR_\alpha(-sd(\boldsymbol{X}(k), \mathcal{O}_\iota)) < \epsilon, \forall k \in \underline{T_f}, \forall \iota \in \underline{N_o}, \tag{8b}$$

$$f_l(\wp_k) < \varepsilon, \forall k \in \underline{T_f}, \tag{8c}$$

where T_f denotes the terminal time step with $\wp_{T_f} = \wp_{targ}$. The optimization objective is to minimize the total transport distance quantified by the weighted sum of Wasserstein distance between consecutive GMMs from $\wp_0$ to $\wp_{targ}$ with corresponding weight coefficients $\lambda_k > 0$. The constraint eq. (8b) leverages SDF to quantify the distance between swarm GMM and an obstacle, and sets an upper bound $\epsilon \in \mathbb{R}$ of the GMM collision risk measured by CVaR. The constraint eq. (8c) employs a linear function $f_l(\cdot)$ to compute the probability values of the swarm PDF at specific spatial positions and sets an upper bound $\varepsilon \in (0, 1)$ to avoid excessive clustering of robots. The proposed (P0) represents an infinite-dimensional optimization problem, rendering it computationally prohibitive.

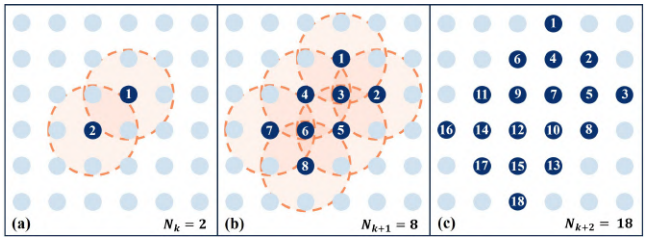

Fig. 3. Example of determining $\mathcal{GC}_k$ and N_k, $\forall k \in \underline{T_f}$. (a), (b), and (c) represent GMMs at time step k, $k+1$, and $k+2$, respectively. Light blue circles depict GCs in $\mathcal{C}$ and deep blue circles with numbers represent the GCs at specific time step and their corresponding indices. Light orange circles denote the transport radius among GCs between adjacent time steps.

We propose to reduce the complexity of (P0) with spatial discretization and turning (P0) into an MPC formulation. In particular, we establish a set $\mathcal{C} = \{g_c^j | g_c^j = \mathcal{N}(\boldsymbol{\mu}_c^j, \boldsymbol{\Sigma}_c^j), \forall j \in \underline{N_c}\}$ consisting of N_c GCs whose mean positions are uniformly distributed in the planning space and covariance matrices are defined concerning the obstacle layout, as will be discussed in Sec. V-B. All GCs of the swarm GMMs in the planning stage are assumed to be chosen only from $\mathcal{C}$. We then design a GC transport radius considering the robot speed limit. Weight transfer among GCs at adjacent time steps is permitted only within this specified transport radius. Thus, the set of all GCs at time step k denoted as $\mathcal{GC}_k$ and the number of GCs at kth step denoted as $N_k = |\mathcal{GC}_k|$ can be predetermined. The GMM trajectory planning problem in (P0) is therefore transformed into determining the weights of the corresponding $\mathcal{GC}_k$. A schematic explanation is given in Fig. 3. Problem (P0) is thus reformulated into an MPC problem (P) as follows,

$$\min_{\boldsymbol{\pi}} \sum_{p=1}^{h} \sum_{i=1}^{N_{k+p-1}} \sum_{j=1}^{N_{k+p}} \lambda_{k+p-1} W_2^2(g_{k+p-1}^i, g_{k+p}^j) \pi_{k+p-1}(i,j) + \sum_{m=1}^{N_{k+h}} \sum_{n=1}^{N_{targ}} \lambda_{k+h} \mathcal{Q}_{k+h}(m,n) \pi_{k+h}(m,n) \tag{9a}$$

$$\text{s.t.} \quad \omega_{k+p-1}^i = \sum_{j=1}^{N_{k+p}} \pi_{k+p-1}(i,j), \ \forall i \in \underline{N_{k+p-1}}, \forall p \in \underline{h} \tag{9b}$$

$$\omega_{k+p}^j = \sum_{i=1}^{N_{k+p-1}} \pi_{k+p-1}(i,j), \ \forall j \in \underline{N_{k+p}}, \forall p \in \underline{h} \tag{9c}$$

$$\omega_{k+h}^m = \sum_{n=1}^{N_{targ}} \pi_{k+h}(m,n), \ \forall m \in \underline{N_{k+h}} \tag{9d}$$

$$\omega_{targ}^{n} = \sum_{m=1}^{N_{k+h}} \pi_{k+h}(m,n), \forall n \in \underline{N_{targ}} \tag{9e}$$

$$\sum_{m=1}^{N_{k+p}} \omega_{k+p}^{m} = 1, \sum_{n=1}^{N_{targ}} \omega_{targ}^{n} = 1, \forall p \in \underline{h} \tag{9f}$$

$$\boldsymbol{\omega}_{k+p} \geqslant \mathbf{0}, \boldsymbol{\omega}_{targ} \geqslant \mathbf{0}, \forall p \in \underline{h} \tag{9g}$$

$$\sum_{i=1}^{N_{k+p-1}} \sum_{j=1}^{N_{k+p}} f_l(g_{k+p}^{j}) \pi_{k+p-1}(i,j) < \varepsilon, \forall p \in \underline{h} \tag{9h}$$

$$CVaR_{\alpha}(-sd(\boldsymbol{X}(k+p), \mathcal{O}_{\iota})) < \epsilon, \forall \iota \in \underline{N_o}, \forall p \in \underline{h} \tag{9i}$$

where the optimization variable is the transport policy $\boldsymbol{\pi} = [\pi_k(\cdot,\cdot), \pi_{k+1}(\cdot,\cdot), \cdots, \pi_{k+h}(\cdot,\cdot)]$, comprising a stack of joint Probability Mass Functions (PMFs) whose marginal PMFs are denoted by eqs. (9b) to (9e). As an example, $\pi_k(i,j)$ represents the weight transported from g_k^i to g_{k+1}^j. The former half of the objective function (eq. (9a)) is the stage cost incurred within the MPC planning horizon with weights $\lambda_{k+p-1}, \forall p \in \underline{h}$, while the latter half represents the terminal cost generated from the final step of the planning horizon to T_f with the weight λ_{k+h}. Based on the design of $\mathcal{C}$, $W_2(\cdot,\cdot)$ in the stage cost can be calculated offline. For unit terminal cost denoted by $\mathcal{Q}_{k+h}(m,n)$, we construct a directed graph $\mathcal{DG} = (\mathcal{V}, \mathcal{E})$, where the node set $\mathcal{V}$ consists of the mean positions of GCs, and the weights on the edges $\mathcal{E}$ represent the transport costs. Then $\mathcal{Q}_{k+h}(m,n)$ is obtained by applying a Dijkstra's algorithm on $\mathcal{DG}$. Equation (9h) is the discretized form of eq. (8c), which is a linear function. Therefore, all parts in (P) except the constraint eq. (9i) are linear with respect to (w.r.t.) $\boldsymbol{\pi}$. The formulation of constraint eq. (9i) will be elaborated in Sec. V-A. The optimal swarm PDF at the next time step is obtained by

$$\wp_{k+1}^{*} = \sum_{j=1}^{N_{k+1}} \sum_{i=1}^{N_k} \pi_k^{*}(i,j) g_{k+1}^{j}, \tag{10}$$

where $\pi_k^{*}(i,j)$ is obtained from $\boldsymbol{\pi}^{*}$ by solving (P). A series of intermediate GMMs is then constructed via linear interpolation between $\wp_k$ and $\wp_{k+1}^{*}$, ensuring a smooth GMM transformation between two adjacent time steps [6].

B. GMM Estimation from Robot Positions

At the beginning of each GMM planning phase, a GMM representation of the swarm is needed. The initial GMM $\wp_0$ is obtained by Expectation-Maximization (EM) algorithm based on the robots' initial positions. At the time step $k > 0$, the last planning result $\wp_k^{*}$ is considered as the desired GMM of the current robot swarm. As some robots may fail to follow $\wp_k^{*}$, we propose Alg. 1 to establish the current swarm GMM $\wp_k^{est}$ based on robots' actual positions, $\wp_k^{*}$ and $\mathcal{C}$.

Alg. 1 first identifies robots that deviate significantly from the desired GMM. A robot is considered to be an outlier when the PDF value at the robot position of $\wp_k^{*}$ is less than a user-defined threshold η (Alg. 1-Alg. 1). Alg. 1 then matches each outlier robot to a GC. Specifically, Alg. 1 to Alg. 1 compute the PDF value of every GC in $\mathcal{C}$ at an outlier robot's position. The GC of the highest PDF value is considered as the GC representing the outlier robot (Alg. 1). For the GMM representation of the outlier robots at the kth time step $\wp_k^{out}$, every outlier robot contributes equally to the weight of GC, from which the robot is most likely to originate (Alg. 1). Alg. 1 then generates $\wp_k^{out}$. Ultimately, we merge $\wp_k^{out}$ representing robot outliers and $\wp_k^{*}$ representing robot inliers, with relative weights proportional to respective group sizes (Alg. 1).

Algorithm 1 EstGMM

Input: $\mathcal{C}$, robot positions $[\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{N_r}]$, $\wp_k^{*}$, η
Output: $\wp_k^{est}$

Initialization: $\boldsymbol{\omega}_c = [\omega_1, \omega_2, \cdots, \omega_{N_c}] \leftarrow \mathbf{0}$, number of outlier robots $N_{ro} = 0$
for $i = 1$ to N_r **do**
  $p_i = f_{\wp_k^{*}}(\boldsymbol{x}_i)$, where $f_{\wp_k^{*}}(\cdot)$ denotes PDF of $\wp_k^{*}$
  if $p_i < \eta$ **then**
    $N_{ro} = N_{ro} + 1$
    for $j = 1$ to N_c **do**
      $p_j = f_{g_c^j}(\boldsymbol{x}_i)$, where $f_{g_c^j}(\cdot)$ is PDF of g_c^j
    end for
    $q = \arg\max_{j \in \underline{N_c}}(p_j)$
    Modify the qth element of $\boldsymbol{\omega}_c$: $\omega_q \leftarrow \omega_q + \frac{1}{N_r}$
  end if
end for
Obtain $\wp_k^{out}$ by selecting the non-zero elements from $\boldsymbol{\omega}_c$ along with their corresponding GCs from $\mathcal{C}$
Obtain $\wp_k^{est}$ by merging $\wp_k^{*}$ and $\wp_k^{out}$ with their relative weights $(N_r - N_{ro})/N_r$ and N_{ro}/N_r

C. Microscopic Control Stage

This section covers the distributed microscopic control that allows the robots to both keep up with the GMM obtained from macroscopic planning stage and avoid collision. The intermediate waypoints of each robot are determined using the APF method (Sec. IV-C.1), followed by our proposed DMPC to enable waypoint tracking (Sec. IV-C.2).

1) Intermediate Waypoint Generation: We propose to determine the intermediate waypoints of robots based on the series of interpolated GMMs between $\wp_k$ and $\wp_{k+1}^{*}$ by sequentially using the designed APF policy as follows,

$$\begin{aligned} &\boldsymbol{x}_i^{ref}(s+1) = \boldsymbol{x}_i^{ref}(s) - \psi \cdot \nabla_{\boldsymbol{x}_i^{ref}(s)} U(s), \\ &s = 0, 1, \cdots, N_k^{inter}, \forall i \in \underline{N_r}, \end{aligned} \tag{11}$$

where $\boldsymbol{x}_i^{ref}(s)$ denotes the reference point at the sth interpolated GMM and $\boldsymbol{x}_i^{ref}(0) = \boldsymbol{x}_i(k)$. The number of interpolated GMMs between $\wp_k$ and $\wp_{k+1}^{*}$ is denoted by N_k^{inter}, and we obtain $\boldsymbol{x}_i(k+1) = \boldsymbol{x}_i^{ref}(N_k^{inter}+1)$. For the APF construction, $\psi > 0$ denotes the step size, and $U(s)$ is the sum of the attractive potential field $U_{att}(s)$ and the repulsive potential field $U_{rep}(s)$, i.e. $U(s) = U_{att}(s) + U_{rep}(s)$.

Specifically, $U_{att}(s)$ is proportional to the mean squared error between the current GMM estimated from robot positions using Kernel Density Estimation (KDE) and the next interpolated GMM. The $U_{rep}(s)$ consists of two components,

which are inversely proportional to the inter-robot distance and the distance between the robot and obstacles, respectively. More details of the APF construction can be found in [6]. The APF can thus keep track of $\wp^*_{k+1}$ and avoid collisions.

2) Distributed Goal Tracking with DMPC: We design a DMPC for robots to track the intermediate waypoints given by APF. The optimization problem for the ith robot can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{u}_i} \ & Q^i = Q^i_{\text{track}} + Q^i_{\text{control}} \\ \text{s.t.} \ & C^i_{\text{dynamic}} = 0,\ C^i_{\text{collision}} \leqslant 0, \end{aligned} \tag{12}$$

where the optimization variable $\boldsymbol{u}_i$ represents the control sequence of the ith robot over the predictive interval. The objective is to track the intermediate waypoints generated by the APF while minimizing the energy consumption caused by the control inputs. The objective function Q^i consists of Q^i_{track} penalizing the squared trajectory tracking error and Q^i_{control} penalizing the control inputs. In terms of the constraints, C^i_{dynamic} represents the robot's kinematic model, while the inequality constraint $C^i_{\text{collision}}$ involves robot-obstacle collision avoidance using collision-free polytopes proposed by [19] and inter-robot collision avoidance with ORCA [22]. These constraints are linearized, thus transforming eq. (12) into a Quadratic Programming (QP) problem that can be solved with high computational efficiency.

V. Swarm Collision Risk Awareness

A. GMM Collision Risk Quantification with CVaR

We derive an analytical expression for the CVaR of a GMM and propose a GMM collision risk quantification approach in this subsection.

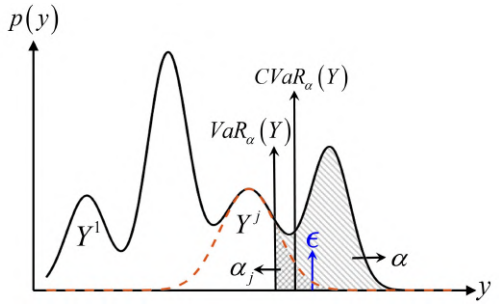

Fig. 4. The graphical interpretation of the CVaR computation for a GMM. For each GC Y_j, the corresponding α_j is calculated as the integral of the PDF of Y^j over the interval $[VaR_\alpha(Y), +\infty)$, and α is computed as the integral of the PDF of Y over the interval $[VaR_\alpha(Y), +\infty)$. The CVaR of Y is calculated as (16), where ϵ serves as the upper bound of $CVaR_\alpha(Y)$.

We first derive the stochastic SDF between Gaussian distributions and an environmental obstacle (Lemma 1), and then extend the result to SDF for GMMs through mixture properties (Lemma 2). Next, we derive an analytical CVaR expression for GMMs (Theorem 1), building upon which eq. (9i) is transformed into the analytical form of eq. (16).

We utilize SDF denoted as $sd(\boldsymbol{X}, \mathcal{O}_\iota)$ to characterize the distance between swarm state $\boldsymbol{X} \sim \sum_{j=1}^N \omega^j \mathcal{N}(\boldsymbol{\mu}^j, \boldsymbol{\Sigma}^j)$ and an obstacle $\mathcal{O}_\iota, \iota \in \underline{N_o}$. We first propose the following lemma regarding the SDF distribution of a Gaussian random variable:

Lemma 1. [Stochastic SDF of a Gaussian Random Variable] *For the jth GC of the GMM denoted as $\boldsymbol{X}^j \sim \mathcal{N}(\boldsymbol{\mu}^j, \boldsymbol{\Sigma}^j)$, the SDF r.v. $sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$ between $\boldsymbol{X}^j$ and $\mathcal{O}_\iota$ can be approximated as a Gaussian distribution*

$$sd(\boldsymbol{X}^j, \mathcal{O}_\iota) \sim \mathcal{N}(sd(\boldsymbol{\mu}^j, \mathcal{O}_\iota), (\boldsymbol{n}^j)^T \boldsymbol{\Sigma}^j \boldsymbol{n}^j), \tag{13}$$

where $sd(\boldsymbol{\mu}^j, \mathcal{O}_\iota)$ is the deterministic SDF between $\boldsymbol{\mu}^j$ and $\mathcal{O}_\iota$, and $\boldsymbol{n}^j$ is the corresponding normal vector.

The proof is given in Appendix I. We further derive the relation between $sd(\boldsymbol{X}, \mathcal{O}_\iota)$ and $sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$ in Lemma 2:

Lemma 2. [Stochastic SDF of a GMM Random Variable] *The r.v. $sd(\boldsymbol{X}, \mathcal{O}_\iota)$ and $\sum_{j=1}^N \omega^j sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$ follow the same distribution. Therefore, $sd(\boldsymbol{X}, \mathcal{O}_\iota)$ can be approximated as a GMM distribution*

$$sd(\boldsymbol{X}, \mathcal{O}_\iota) \sim \sum_{j=1}^N \omega^j \mathcal{N}(sd(\boldsymbol{\mu}^j, \mathcal{O}_\iota), (\boldsymbol{n}^j)^T \boldsymbol{\Sigma}^j \boldsymbol{n}^j). \tag{14}$$

The proof is given in Appendix II. We quantify the collision risk between swarm GMM and obstacles, denoted by Y, with the inverse of SDF, i.e, $Y = -sd(\boldsymbol{X}, \mathcal{O}_\iota)$, which then follows a GMM. We derive an analytical expression for the CVaR of a GMM r.v. in the following theorem:

Theorem 1. [CVaR of a GMM Random Variable] *The CVaR of a GMM r.v. Y at risk acceptance level α can be represented by the summation of CVaRs of the corresponding Gaussian component Y^j of Y at risk acceptance level α_j:*

$$CVaR_\alpha(Y) = \frac{1}{\alpha} \sum_{j=1}^N \omega^j \alpha_j CVaR_{\alpha_j}(Y^j), \tag{15}$$

where ω^j denotes the weight of Y^j, and $\alpha_j = \int_{VaR_\alpha(Y)}^{+\infty} f_{Y^j}(y^j) dy^j$ is the tail probability of Y^j distribution at the $VaR_\alpha(Y)$-th quantile with $f_{Y^j}(\cdot)$ denoting the PDF of Y^j. Besides, the following relationship between α and α_j holds, $\alpha = \sum_{j=1}^N \omega_j \alpha_j$.

The proof is given in Appendix III. Based on Theorem 1, the constraint eq. (9i) can be analytically expressed as follows,

$$\begin{aligned} & CVaR_\alpha(Y(k+p)) \\ & = \frac{1}{\alpha} \sum_{j=1}^{N_{k+p}} \omega^j_{k+p} \alpha_j \, CVaR_{\alpha_j}(Y^j(k+p)) < \epsilon, \forall p \in \underline{h}. \end{aligned} \tag{16}$$

Fig. 4 illustrates the CVaR of a GMM. While $CVaR_{\alpha_j}(Y^j(k+p))$ employs an analytical expression (eq. (7)), the α_j associated with $VaR_\alpha(Y(k+p))$ is an implicit function of the weights of GCs, making eq. (16) the sole nonlinear constraint in (P). In Sec. VI, we introduce a computationally efficient sequential quadratic algorithm for (P) by exploiting the concavity of CVaR in eq. (16).

B. Risk-Aware Planning Space Discretization

We present a method for discretizing the planning space considering the collision risk, specifically addressing the construction of the set $\mathcal{C}$ as outlined in Sec. IV-A.

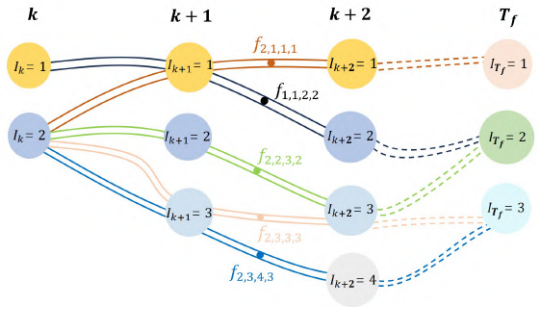

Fig. 5. Illustration of the linear relationship among the variables $\boldsymbol{\omega}$, $\boldsymbol{\pi}$, and $\boldsymbol{F}$. The figure provides an example with planning horizon $h = 2$, where each circle represents a GC at a specific time step, and circles with the same color denote the same GC across various time steps. The colored flow lines represent different flow variables $f_{I_k,I_{k+1},I_{k+2},I_{T_f}}$, which are elements of flow vector $\boldsymbol{F}$, to indicate the transition of weights between GCs over time. In this example, each intermediate weight ω is obtained by summing the flow variables passing through a specific GC at a time step, e.g. $\omega^1_{k+1} = f_{1,1,2,2} + f_{2,1,1,1}$. The transport policy π is obtained by summing the flow variables passing through two same GCs betweeen adjacent time steps, e.g. $\pi_k(2,3) = f_{2,3,3,3} + f_{2,3,4,3}$.

To comprehensively cover the workspace, the means of the GCs are uniformly distributed throughout the planning space. As the covariance matrices of these GCs determine the swarm's formation, we propose to minimize the collision risk between GCs and environmental obstacles measured by CVaR, thereby reducing the robots' collision risk with obstacles. Defining $\boldsymbol{\Sigma}_c^j = \begin{bmatrix} \sigma_1^j & \sigma_3^j \\ \sigma_3^j & \sigma_2^j \end{bmatrix}$, we propose an optimization problem as follows,

$$\min_{\boldsymbol{\Sigma}_c^j} \quad \sum_{\iota=1}^{N_o} \lambda_\iota CVaR_\alpha(-sd(\boldsymbol{X}_c^j, \mathcal{O}_\iota)) \tag{17a}$$

$$\text{s.t.} \quad \boldsymbol{\Sigma}_c^j \succ 0 \tag{17b}$$

$$lb_q < \sigma_q^j < ub_q, q \in \{1,2,3\}, \tag{17c}$$

where $\boldsymbol{X}_c^j \sim g_c^j$, $sd(\boldsymbol{X}_c^j, \mathcal{O}_\iota)$ is given by Lemma 2, and CVaR can be calculated using eq. (7). The objective function aims to minimize the collision risk represented by the weighted sum of CVaRs calculated from SDF distributions between the GC and various obstacles with $\lambda_\iota > 0$ denoting their respective weights. Closer obstacles are assigned larger weights. The constraint eq. (17c) is designed to ensure that $\boldsymbol{\Sigma}_c^j$ is far from singularity by choosing appropriate lb_q and ub_q. Based on eq. (7), eq. (17a) is formulated as

$$\min_{\boldsymbol{\Sigma}_c^j} \quad -\sum_{\iota=1}^{N_o} \lambda_\iota sd(\boldsymbol{\mu}_c^j, \mathcal{O}_\iota)) + A \sum_{\iota=1}^{N_o} \lambda_\iota \sqrt{\boldsymbol{n}_{j\iota}^T \boldsymbol{\Sigma}_c^j \boldsymbol{n}_{j\iota}}, \tag{18}$$

where $A = \phi(\Phi^{-1}(1-\alpha))/\alpha$ and $\boldsymbol{n}_{j\iota}$ denotes normal vector corresponding to $sd(\boldsymbol{\mu}_c^j, \mathcal{O}_\iota)$. By removing the parts irrelevant to $\boldsymbol{\Sigma}_c^j$, eq. (18) can be written as

$$\min_{\boldsymbol{\Sigma}_c^j} \quad \sum_{\iota=1}^{N_o} \lambda_\iota \sqrt{\boldsymbol{n}_{j\iota}^T \boldsymbol{\Sigma}_c^j \boldsymbol{n}_{j\iota}}, \tag{19}$$

which is concave w.r.t. $\boldsymbol{\Sigma}_c^j$, rendering high computational complexity. Applying mean inequality to eq. (19), we obtain

$$\sum_{\iota=1}^{N_o} \lambda_\iota \sqrt{\boldsymbol{n}_{j\iota}^T \boldsymbol{\Sigma}_c^j \boldsymbol{n}_{j\iota}} \leqslant \sqrt{N_o \sum_{\iota=1}^{N_o} \lambda_\iota^2 \boldsymbol{n}_{j\iota}^T \boldsymbol{\Sigma}_c^j \boldsymbol{n}_{j\iota}}. \tag{20}$$

Incorporating a logarithmic regularization term of the determinant of $\boldsymbol{\Sigma}_c^j$ into the square of the right-hand side of eq. (20), we propose the following convex optimization problem:

$$\min_{\boldsymbol{\Sigma}_c^j} \quad \kappa N_o \sum_{\iota=1}^{N_o} \lambda_\iota^2 \boldsymbol{n}_{j\iota}^T \boldsymbol{\Sigma}_c^j \boldsymbol{n}_{j\iota} - \log\det(\boldsymbol{\Sigma}_c^j) \tag{21a}$$

$$\text{s.t.} \quad \boldsymbol{\Sigma}_c^j \succ 0 \tag{21b}$$

$$lb_q < \sigma_q^j < ub_q, q \in \{1,2,3\}, \tag{21c}$$

where $\kappa > 0$ is the weight coefficient that balances two objectives, $\det(\cdot)$ denotes the determinant function. Compared with the original optimization problem given by eq. (17), eq. (21) not only minimizes the GC's collision risks but also maximizes the dispersion of robot swarm represented by $\det(\boldsymbol{\Sigma}_c^j)$, providing the swarm with more flexibility in formations.

VI. Online Solution of Non-Convex MPC

In this section, we tackle the computational challenges of (P). The CVaR of a GMM random variable is concave w.r.t. $\boldsymbol{\omega}$ [37]. Due to the linear relation between $\boldsymbol{\omega}$ and $\boldsymbol{\pi}$ (eqs. (9b) to (9e)), constraint eq. (9i) is also concave w.r.t. $\boldsymbol{\pi}$. The proposed (P) constitutes a special class of non-convex optimization problems, characterized by a single concave constraint, while the objective function and all remaining constraints are linear. In this section, we exploit this structure and propose the algorithm OptiGMM with guaranteed convergence to solve (P), by sequentially solving a set of quadratic programming subproblems.

A. Concave Constraint Linearization

We propose to linearize eq. (9i) to achieve a convex approximation. However, the gradient of eq. (9i) w.r.t $\boldsymbol{\pi}$ is difficult to compute. We first transform the decision variable $\boldsymbol{\pi}$ into *Flow Vector* $\boldsymbol{F} \in \mathbb{R}_+^{N_{targ}\prod_{i=0}^{h} N_{k+i}}$ (Fig. 5). Each element of $\boldsymbol{F}$ is a *Flow Variable* $f_{I_k,I_{k+1},\cdots,I_{k+h},I_{targ}} \in \mathbb{R}_+$, determined by a set of ordered index variables $I_k \in \underline{N_k}, \cdots, I_{k+h} \in \underline{N_{k+h}}, I_{targ} \in \underline{N_{targ}}$, which represents the GMM weight "flowing" along a specific route noted as $g_k^{I_k}, \cdots, g_{k+h}^{I_{k+h}}, g_{targ}^{I_{targ}}$. For notational simplicity, we omit the subscripts of a flow variable and briefly denote it as f.

There exists a linear relationship among $\boldsymbol{\omega}$, $\boldsymbol{\pi}$ and $\boldsymbol{F}$. The sum of all flow variables that pass through the jth GC at a given time step yields the weight of GC at that step, i.e.,

$$\omega_{k+p}^j = \sum_{I_{k+p}=j} f, \ \forall p \in \underline{h}, \quad \omega_{targ}^j = \sum_{I_{targ}=j} f. \tag{22}$$

Furthermore, the optimization variable $\boldsymbol{\pi}(\cdot,\cdot)$ presented in (P) can also be transformed into the sum of all the flow variables passing through two particular GCs at consecutive time steps:

$$\pi_{k+p-1}(m,n) = \begin{cases} \sum\limits_{I_{k+p-1}=m, I_{k+p}=n} f, & \forall p \in \underline{h} \\ \sum\limits_{I_{k+h}=m, I_{targ}=n} f, & p = h+1 \end{cases}. \quad (23)$$

Algorithm 2 OptiGMM

Input: a feasible solution $\hat{\boldsymbol{F}}^v, \mathcal{GC}_{k+1:h}, \alpha, \eta > 0, \mathcal{O} = \bigcup_{\iota=1}^{N_o} \mathcal{O}_\iota$
Output: $\wp^*_{k+1}$
Initialization: $v \leftarrow 0$
loop
  for $\iota = 1$ to N_o **do**
    for $p = 1$ to h **do**
      compute $\boldsymbol{\omega}_{k+p}$ from $\hat{\boldsymbol{F}}^v$
      compute $VaR_\alpha(Y(k+p))$ with the bisection method
      for $j = 1$ to N_{k+p} **do**
        compute α_j with $\mathbf{ccdf}_{Y^j}(VaR_\alpha(Y(k+p)))$
      end for
      compute $CVaR_\alpha(Y(k+p))|_{\boldsymbol{F}^v=\hat{\boldsymbol{F}}^v}$
      compute $\nabla_{\boldsymbol{F}^v} CVaR_\alpha(Y(k+p))\big|_{\boldsymbol{F}^v=\hat{\boldsymbol{F}}^v}$
      linearize $CVaR_\alpha(Y(k+p))$ at $\boldsymbol{F}^v = \hat{\boldsymbol{F}}^v$
    end for
  end for
  solve (P^v) to find $\boldsymbol{F}^{v*}$
  $\hat{\boldsymbol{F}}^{v+1} \leftarrow \hat{\boldsymbol{F}}^v + \gamma^v(\boldsymbol{F}^{v*} - \hat{\boldsymbol{F}}^v)$, where $\gamma^v \in (0,1]$
  if $\|\boldsymbol{F}^{v*} - \hat{\boldsymbol{F}}^v\| < \eta$ **then**
    break
  end if
  $v \leftarrow v+1$
end loop
Compute $\wp^*_{k+1}$ from $\boldsymbol{F}^{v*}$ and $\mathcal{GC}_{k+1}$ based on eq. (10) and eq. (23)

To compute the gradient of eq. (9i) w.r.t. $\boldsymbol{F}$, we first derive the analytical gradient of CVaR w.r.t. $\boldsymbol{\omega}$ in the following proposition.

Proposition 1. *Given a risk acceptance level α and a GMM r.v. Y, $CVaR_\alpha(Y)$ is differentiable w.r.t the GC weight vector $\boldsymbol{\omega} = [\omega^1, \omega^2, \cdots, \omega^N]$, and the gradient of $CVaR_\alpha(Y)$ is*

$$\nabla_{\boldsymbol{\omega}} CVaR_\alpha(Y) = \left[\frac{\partial CVaR_\alpha(Y)}{\partial \omega^1}, \cdots, \frac{\partial CVaR_\alpha(Y)}{\partial \omega^N}\right]^T, \quad (24)$$

where

$$\frac{\partial CVaR_\alpha(Y)}{\partial \omega^j} = \frac{\alpha_j}{\alpha} CVaR_{\alpha_j}(Y^j) - \frac{\alpha_j}{\alpha} VaR_\alpha(Y), \forall j \in \underline{N}. \quad (25)$$

The proof is presented in Appendix IV. Based on the chain rule, the gradient of eq. (9i) w.r.t. $\boldsymbol{F}$, denoted as $\nabla_{\boldsymbol{F}} CVaR_\alpha(Y(k+p))$, can be obtained with each partial derivative from the gradient vector determined by

$$\frac{\partial CVaR_\alpha(Y(k+p))}{\partial f} = \frac{\partial CVaR_\alpha(Y(k+p))}{\partial \omega^j_{k+p}}, \forall p \in \underline{h}, \quad (26)$$

for all the flow variables f with $I_{k+p} = j$. The constraint eq. (9i) can be further linearized leveraging the first-order Taylor expansion at a feasible solution of (P), $\hat{\boldsymbol{F}}$, as

$$CVaR_\alpha(Y(k+p)) = CVaR_\alpha(Y(k+p))|_{\boldsymbol{F}=\hat{\boldsymbol{F}}} + \nabla^T_{\boldsymbol{F}} CVaR_\alpha(Y(k+p))|_{\boldsymbol{F}=\hat{\boldsymbol{F}}}(\boldsymbol{F} - \hat{\boldsymbol{F}}). \quad (27)$$

B. Sequential Optimization Algorithm

We introduce a strongly convex approximation of the objective function of (P), thereby proposing the following subproblem (P^v) with v denoting the iteration index. Given an initial solution of (P^v), $\hat{\boldsymbol{F}}^v$, which is also a feasible solution of (P), (P^v) is defined as below:

$$\min_{\boldsymbol{F}^v} \quad J + \frac{\tau}{2}\|\boldsymbol{F}^v - \hat{\boldsymbol{F}}^v\|^2 \quad (28a)$$

$$\text{s.t.} \quad (9b) - (9h), (23), \quad (28b)$$

$$eq.\ (27) < \epsilon, \forall p \in \underline{h}, \quad (28c)$$

where J represents the objective function of (P) defined in eq. (9a), $\tau > 0$ is the regularization weight, and eq. (28c) is calculated at $\hat{\boldsymbol{F}}^v$ based on eq. (27). We then propose a sequential algorithm OPTIGMM to address the non-convex MPC (P) by sequentially solving (P^v), $v \in \mathbb{N}_+$. Each subproblem (P^v) is a QP problem, which enables computationally efficient solution.

The detailed execution of OPTIGMM is presented in Alg. 2. Every iteration involves first the acquisition of

Algorithm 3 C-ROVER

Input: Workspace $\mathcal{W}$, obstacles $\mathcal{O}$, desired target distribution $\wp_{targ}$, initial robot positions $\{x_i(0)\}_{i=1}^{N_r}$
Output: robot trajectories $\{x_i(0{:}T_f)\}$ and final swarm GMM $\wp_{T_f}$
Risk-aware discretization: Build the GC set $\mathcal{C}$ over $\mathcal{W}$ and its graph
Initialization: Estimate initial swarm GMM $\wp_0$ from $\{x_i(0)\}_{i=1}^{N_r}$ with *EM* Algorithm; $k = 0$
while robots have not reached the target area **do**
  Solve the risk-aware MPC with *OPTIGMM* Algorithm to obtain the transport policy π^*_k and the next-step GMM $\wp^*_{k+1}$
  Interpolate between $\wp_k$ and $\wp^*_{k+1}$ to form intermediate GMMs
  Use APF to generate waypoints according to the intermediate GMMs
  for each robot i **do**
    Run DMPC to track its individual waypoints, and update its position to $x_i(k{+}1)$ by applying $u_i(k)$
  end for
  Collect $\{x_i(k{+}1)\}_{i=1}^{N_r}$ and update the current swarm GMM $\wp_{k+1}$ via *EstGMM* Algorithm
  $k \leftarrow k{+}1$
end while
$T_f = k$

GMM $Y(k+p)$ with the knowledge of $\boldsymbol{\omega}_{k+p}$ (Alg. 2) and $\mathcal{GC}_{k+p}$. Alg. 2 to Alg. 2 encompass the linearization of $CVaR_\alpha(Y(k+p))$. Intermediate variable $VaR_\alpha(Y(k+p))$ is calculated in Alg. 2 with a bisection method between the upper and lower bounds established in Proposition 2. Alg. 2 computes $\alpha_{1:N_{k+p}}$ leveraging the complementary cumulative distribution function (CCDF) denoted by $\mathbf{ccdf}_{Y^j}(\cdot)$. We then compute $CVaR_\alpha(Y(k+p))$ and its gradient in Alg. 2, and obtain the linearized $CVaR_\alpha(Y(k+p))$ constraint in Alg. 2. Alg. 2 to Alg. 2 include solving (P^v) to find the optimal iterative solution $\boldsymbol{F}^{v*}$ and moving $\hat{\boldsymbol{F}}^v$ towards $\boldsymbol{F}^{v*}$ to obtain $\hat{\boldsymbol{F}}^{v+1}$ with a step size γ^v. This algorithm is sequentially executed until $\boldsymbol{F}^v$ meets the stationary condition of (P). Then the GMM at the next time step $\wp^*_{k+1}$ is computed in Alg. 2.

Proposition 2. *Given a risk acceptance level α, the VaR of a GMM r.v. Y is bounded by the maximum value and the minimum value of the VaR of the corresponding GCs Y^j under the same α, denoted by:*

$$\min_{j=1:N} VaR_\alpha(Y^j) \leqslant VaR_\alpha(Y) \leqslant \max_{j=1:N} VaR_\alpha(Y^j). \quad (29)$$

The proof is given in Appendix V.

C. Convergence Analysis of OptiGMM

Recall that the constraint eq. (9i) is concave w.r.t. $\boldsymbol{\omega}$ [37]. This concavity still holds w.r.t. $\boldsymbol{F}$ due to the linear relationship between $\boldsymbol{\omega}$ and $\boldsymbol{F}$ as shown in eq. (22). Thus, the linearization of eq. (9i) w.r.t. $\boldsymbol{F}$ always leads to an upper bound of eq. (9i), which represents a more conservative approximation of the collision avoidance constraint. Therefore, the feasible domain of (P^v) is always a closed, convex, inner approximation of the feasible domain of (P). Furthermore, the objective function of (P^v) is the strongly convex approximation of the one of (P). It can be easily verified that (P) satisfies Linear Independence Constraint Qualification (LICQ) [38]. Consequently, OPTIGMM converges to a stationary solution of (P) as proved in [39]. The flow of the whole algorithm is given in Alg. 3.

VII. SIMULATION AND RESULTS

The workspace $\mathcal{W}$ is a $[0,200] \times [0,160] m^2$ area with static obstacles. In the macroscopic planning stage, the initial swarm GMM consists of four GCs with the same covariance matrix $\boldsymbol{\Sigma} = 100\boldsymbol{I}_2$, where $\boldsymbol{I}_2$ denotes a 2×2 identity matrix. Other GC parameters are $\boldsymbol{\mu}_1 = [25, 35]$, $\omega_1 = 0.1875$, $\boldsymbol{\mu}_2 = [25, 55]$, $\omega_2 = 0.1875$, $\boldsymbol{\mu}_3 = [25, 115]$, $\omega_3 = 0.375$, $\boldsymbol{\mu}_4 = [25, 135]$, $\omega_4 = 0.25$. The target GMM is composed of three GCs with parameters $\boldsymbol{\mu}_1 = [175, 120]$, $\omega_1 = 0.25$, $\boldsymbol{\mu}_2 = [175, 60]$, $\omega_2 = 0.375$, $\boldsymbol{\mu}_3 = [175, 40]$, $\omega_3 = 0.375$ and an identical covariance matrix $\boldsymbol{\Sigma} = 100\boldsymbol{I}_2$. The GC set $\mathcal{C}$ is predefined by taking the mean of each GC on fixed grids, where the X coordinates of the grids range from 5 to 195 and the Y coordinates range from 5 to 155, with an increment interval of 10. Consequently, the set $\mathcal{C}$ comprises a total of $20\times 16 = 320$ GCs. The covariance matrices of the GCs in $\mathcal{C}$ are predetermined via optimizations in Sec. V-B. To specify the optimization problem (P0), α and ϵ are set to 0.05 and 0, respectively. In eq. (9a), we set $h = 2$, $\lambda_{k+h} = 3$, and $\lambda_{k+p-1} = 1$, $\forall p \in \underline{h}$. The η in the convergence condition of Alg. 2 is 10^{-3}. In the microscopic control stage, we set

TABLE II
PERFORMANCE METRICS OF C-ROVER / FC / PC / TUBE / CC-ADOC IN SCENARIO 1

N_r	$t(t_p)$ (min)	$\overline{D}$ (m)	$\min(d_{ij})$ (m)	$\min(d_{io})$ (m)	$\bar{E}$ (J/kg)
25	2.29 (0.25) / 1.82 / 0.81 / 10.68 (0.88) / 6.63(0.40)	183.49 / 273.57 / 168.49 / 240.84 / 228.10	0.65 / 0.84 / 0.21 / 0.31 / 0.32	1.29 / 9.19 / 0.06 / 1.47 / 0.12	1207.72 / 1937.53 / 1593.35 / 1448.90 / 1339.89
50	2.85 (0.25) / 5.83 / 1.96 / NaN / 9.50(1.06)	174.92 / 270.79 / 163.28 / NaN / 215.22	0.28 / 0.10 / 0.30 / NaN / 0.33	1.09 / 9.93 / 0.02 / NaN / 0.12	1137.02 / 1890.54 / 1533.72 / NaN / 1242.83
250	7.28 (0.25) / 25.69 / 23.31 / NaN / 17.93(1.17)	172.78 / 333.25 / 162.42 / NaN / 199.46	0.19 / 0.00 / -0.06 / NaN / 0.17	0.13 / 4.89 / -4.08 / NaN / 0.12	1107.28 / 2365.68 / 1338.30 / NaN / 1143.31
500	12.95 (0.25) / 44.51 / 116.76 / NaN / 32.98(1.63)	172.42 / 279.70 / 162.09 / NaN / 195.96	0.15 / 0.00 / -0.22 / NaN / 0.23	0.13 / 6.88 / -6.69 / NaN / 0.12	1104.02 / 1958.92 / 1213.77 / NaN / 1108.60
1000	25.89 (0.25) / 141.89 / NaN / NaN / 28.73(1.46)	173.46 / 273.90 / NaN / NaN / 251.99	0.13 / 0.00 / NaN / NaN / 0.08	0.12 / 4.66 / NaN / NaN / 0.11	1101.73 / 1908.07 / NaN / NaN / 1407.42

TABLE III
PERFORMANCE METRICS OF C-ROVER / FC / PC / TUBE / CC-ADOC IN SCENARIO 2

N_r	$t(t_p)$ (min)	$\overline{D}$ (m)	$\min(d_{ij})$ (m)	$\min(d_{io})$ (m)	$\bar{E}$ (J/kg)
25	3.43 (0.47) / 0.93 / NaN / 7.14 (0.47) / 6.63(0.68)	197.3 / 250.87 / NaN / 235.7 / 206.77	1.12 / 0.17 / NaN / 0.01 / 1.23	0.84 / 0.21 / NaN / 3.06 / 0.74	1242.63 / 1791.47 / NaN / 1402.01 / 1243.56
50	4.40 (0.47) / 2.82 / NaN / NaN / 9.76(1.08)	197.35 / 271.87 / NaN / NaN / 207.58	0.80 / 0.00 / NaN / NaN / 0.70	0.68 / 0.88 / NaN / NaN / 0.92	1173.97 / 1914.07 / NaN / NaN / 1215.28
250	9.99 (0.49) / 21.26 / NaN / NaN / 28.08(2.05)	179.30 / 270.78 / NaN / NaN / 199.83	0.26 / 0.00 / NaN / NaN / 0.24	0.12 / 0.91 / NaN / NaN / 0.75	1121.46 / 1901.96 / NaN / NaN / 1187.51
500	17.04 (0.47) / 33.81 / NaN / NaN / 48.95(2.49)	180.05 / 274.38 / NaN / NaN / 198.12	0.25 / 0.00 / NaN / NaN / 0.22	0.12 / 0.94 / NaN / NaN / 0.72	1126.29 / 1928.42 / NaN / NaN / 1178.15
1000	33.76 (0.47) / 99.64 / NaN / NaN / NaN	180.28 / 271.63 / NaN / NaN / NaN	0.23 / 0.00 / NaN / NaN / NaN	0.12 / 0.89 / NaN / NaN / NaN	1122.49 / 1909.18 / NaN / NaN / NaN

the discretization time interval Δt to $0.1s$. The predictive horizon of DMPC is set to 16 steps. Robots adopt differential drive motion models with a maximal speed of 1.5m/s, and the radius of each robot is set to 0.12m. All simulations are run in MATLAB on a desktop (13th Intel(R) i7 CPU@2.10GHz), and QP is solved using the Mosek solver.

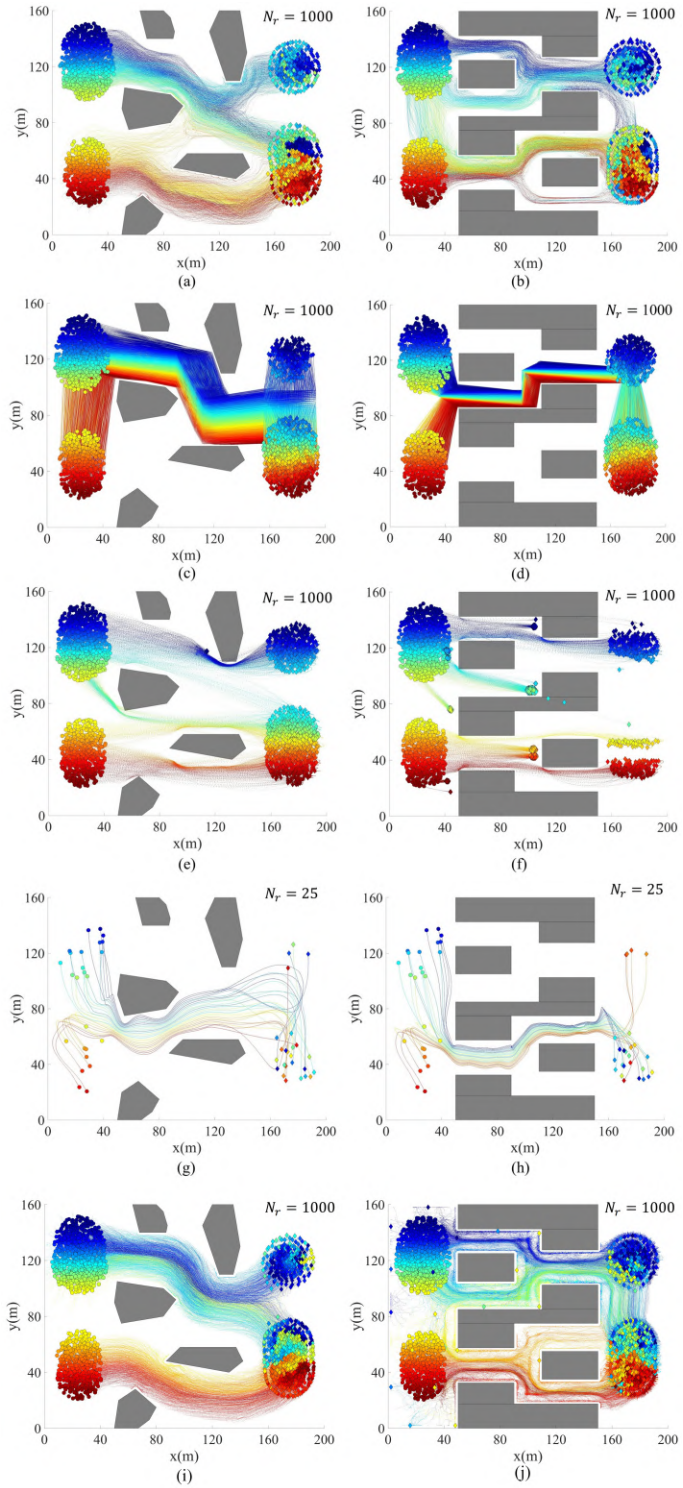

Fig. 6. Figures (a)-(b), (c)-(d), (e)-(f), (g)-(h), and (i)-(j) show the trajectories of robots generated by C-ROVER, FC, PC, Tube, and CC-ADOC, respectively. The initial positions of robots are demonstrated by circles, while their corresponding final positions are highlighted by diamonds. The grey areas denote obstacles. The simulation of FC does not terminate after collisions occur. For the Tube baseline, most robots move outside the workspace when the swarm size exceeds 25. Therefore, results are reported only for the case with 25 robots.

A. *Performance Metrics*

We propose six metrics to evaluate the motion planning performance: (1) The macroscopic planning runtime t_p that includes the GMM estimation phase with Alg. 1 and the GMM planning phase with Alg. 2. (2) The microscopic control runtime t_c that includes the target waypoint generation phase and the DMPC tracking phase. (3) Total runtime $t = t_p + t_c$, and time t set to be infinite when the swarm transport task fails to be completed within a predefined timestep upper bound. (4) Average trajectory length $\bar{D}$ of all robots from the initial area to the target area. (5) Minimum inter-robot distance $min(d_{ij})$ and minimum robot-obstacle distance $min(d_{io})$ maintained by all robots over the whole trajectory. (6) Average per-robot energy consumption $\bar{E}$ (J/kg), denoted as

$$\bar{E}(k) = \frac{\eta}{2N_r} \sum_{i=1}^{N_r} \sum_{\tau=1}^{k} \left[\frac{\|x_i(\tau\Delta t) - x_i[(\tau-1)\Delta t]\|}{\Delta t} \right]^2, \tag{30}$$

where η represents a unit-conversion factor, and Δt is the time step.

The histograms clearly illustrate that α can effectively modulate the proximity of a robot swarm to obstacles. Additionally, we notice a decrease in $\bar{D}$ as α increases, i.e., as α grows from 0.05 to 0.3, $\bar{D}$ decreases from 180.28m to 179.06m. These results can be attributed to the fact that the swarm prefers conservative paths that maintain a greater distance from obstacles and traverse a longer distance to the target area with a smaller α.

B. *Comparison with Benchmark Approaches*

We compare the performance of C-ROVER in different scenarios with several state-of-the-art swarm motion planning approaches, including multi-robot Formation Control (FC) [19], Predictive Control (PC) [11, 12], Tube RRT* (Tube) [40], and Chance-constrained Adaptive Online Distributed Optimal Control (CC-ADOC) [6]. Notice that the

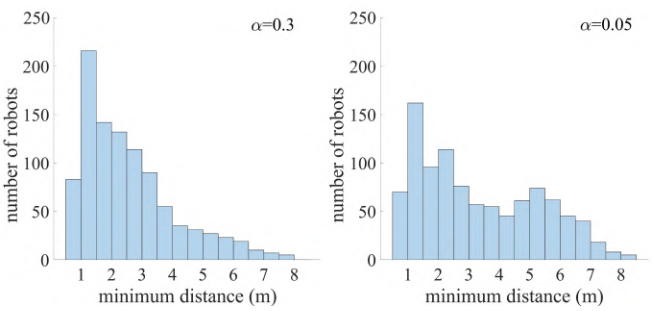

Fig. 7. Histogram of the number of robots in different minimum robot-obstacle distance intervals throughout the entire trajectory, where α equals 0.3 and 0.05, respectively.

global reference trajectories of FC and Tube are obtained by a sampling-based method. Therefore, the evaluation results of FC and Tube are the average values obtained from five independent simulations. As PC can only deal with circular obstacles, the static obstacles are approximated with circular obstacles in the simulation of PC. All approaches are tested with various swarm sizes ($N_r = 25, 50, 250, 500, 1000$) in two typical scenarios. *Scenario 1* consists of large polygonal obstacles and *scenario 2* is more challenging with non-convex obstacles and narrow passages. Scenarios 1 and 2 correspond to the left and right columns in Fig. 6, respectively.

TABLE IV
RISK-AWARENESS SPATIAL DISCRETIZATION WITH 250 ROBOTS

Metrics	$\kappa = 0$	$\kappa = 5$	$\kappa = 10$
$t(t_p)(min)$	33.27 (7.08)	17.38 (0.80)	8.70 (0.30)
$\bar{D}(m)$	22.25	17.43	16.76

1) Trajectory Flexibility and Scalability: A comparison with benchmark approaches in scenario 1 is presented in this subsection. Tube can only accomplish the swarm transport task at a swarm size of 25. As shown in the left column of Fig. 6, C-ROVER achieves high trajectory flexibility, enabling the swarm to split and merge to navigate around obstacles, while the hierarchical baseline FC and Tube confine robots within a formation during the transformation.

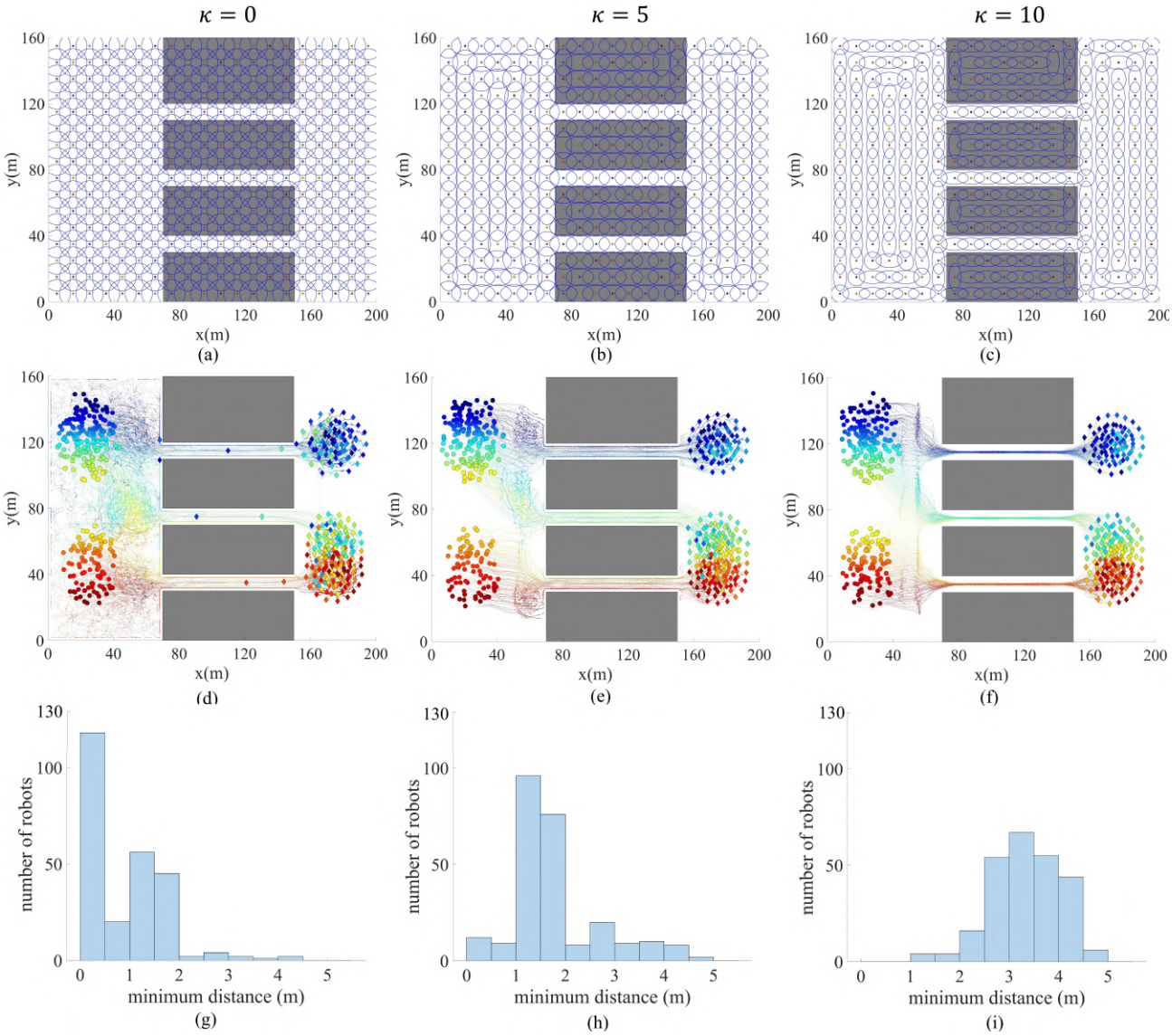

Fig. 8. From left to right, the three columns present the simulation results with 250 robots in a scenario with narrow passages, under the conditions of $\kappa = 0$, $\kappa = 5$, and $\kappa = 10$, respectively. Figures (a) – (c) show the GCs when discretizing the planning space, where the colored filled circles represent the mean positions of the GCs, and the blue hollow circles indicate the 50% probability contours of GCs. Figures (d) – (f) show the trajectories of the robot swarm, where circles denote the initial positions of the robots, and diamonds represent the final positions. Figures (g) – (i) depict histograms of the minimum distances to obstacles for all 250 robots.

The trajectory flexibility of C-ROVER stems from its GMM representation, where adaptive changes in the number and weights of GCs allow the swarm to split and merge as needed.

Tab. II presents a quantitative comparison of different approaches, with NaN indicating the failure to complete the swarm transport task. C-ROVER exhibits the smallest t at large swarm sizes, demonstrating its high scalability. Notably, C-ROVER's t_p remains stable across various swarm sizes, which is expected since the macroscopic planning only depends on the GMM representation rather than the number of robots. Additionally, C-ROVER guarantees safe swarm motions, effectively balancing $min(d_{ij})$ and $min(d_{io})$. In contrast, PC cannot ensure safety in large swarm sizes due to soft collision avoidance constraints. FC and Tube maintain larger $min(d_{io})$ but lead to smaller $min(d_{ij})$ and larger $\bar{D}$ due to formation constraints. Compared with C-ROVER, CC-ADOC does not consider collision severity, allowing the tail of the planned robot distribution to penetrate deeply into obstacles and thereby imposing substantial challenges on the lower-level controller. To ensure safety, the robots often have to take detours, resulting in longer trajectories. Considering trajectory flexibility and computational efficiency, C-ROVER offers the best overall performance.

2) Ability in Navigating Narrow Passages: Scenario 2 highlights the different capabilities of these approaches in navigating narrow passages. Results are given in the right column in Fig. 6 and Tab. III. As PC is designed for circular obstacles, robots will block passage corners and fail to navigate around the obstacles, leading to tasks unfinished. Similar to scenario 1, Tube can only accomplish the swarm transport task when $N_r = 25$. CC-ADOC lacks spatial risk severity distribution modeling, leading to path failures in high-risk regions, so some robots fail to reach the target area. C-ROVER's risk-aware planning space discretization enhances its adaptation to the environment, making it more suitable for navigating narrow passages compared with FC, as reflected by shorter t at large swarm sizes and smaller $\bar{D}$. Metrics $min(d_{ij})$ and $min(d_{io})$ demonstrate the safe motion generated by C-ROVER, FC, and Tube, with C-ROVER achieving more balanced inter-robot and robot-obstacle distances.

C. Risk-Awareness of Motion Planning

1) Risk Management with Different Risk Acceptance Levels: To assess the risk awareness capability of C-ROVER, we compare the swarm behaviour when $\alpha = 0.05$ and 0.3 in scenario 2 with 1000 robots. We plot the histogram of the minimum distances maintained by all robots from obstacles throughout their trajectories (Fig. 7).

2) Risk Management with Risk-Aware Gaussian Components: We demonstrate the importance of considering collision risks from environmental obstacles when discretizing the planning space. In eq. (21a), κ determines the trade-off between swarm collision risk and the dispersion of the robot swarm. We conduct simulations with 250 robots in a scenario containing three extremely narrow passages and analyze the results (Fig. 8 and Tab. IV) for $\kappa = 0$, 5, and 10.

GCs are circular in shape when $\kappa = 0$ (Fig. 8(a)), corresponding to the non-risk-aware spatial discretization. This setting leads to closer proximity to obstacles as shown in Fig. 8(g) compared to other cases. The circular GCs fail to provide the necessary guidance for formation adaptation when the robot swarm enters narrow passages. As a result, the robots tend to disperse near the entrance of the narrow passage due to obstacle avoidance requirements in the microscopic control layer, as illustrated on the left side of Fig. 8(d). The dispersion leads to backward movements, causing an increase in $\bar{D}$, more re-planning iterations, and extended t and t_p. Consequently, the swarm transport task fails to be accomplished within the predefined timestep bound.

For cases with a positive κ, more emphasis is placed on collision risk as κ increases. Both cases finish the swarm transport task. The larger κ facilitates the discrete GCs to adapt their shapes more according to the narrow passages (Fig. 8(b)-Fig. 8(c)), giving the swarm formation guidance better suited for navigation, and thus maintaining a larger distance to obstacles (Fig. 8(h)–Fig. 8(i)). However, a large κ causes the robots to cluster more tightly within the swarm. Therefore, selecting an appropriate κ is essential to balancing safety and efficiency in swarm navigation.

VIII. Real-World Experiments

This section evaluates the real-world performance of C-ROVER with Ackermann autonomous ground vehicles (AGVs), unmanned aerial vehicles (UAVs), and differential-drive robots (DDRs). The experiment setup is given in Fig. 9. In addition to the metrics used in the simulation, we further employ the following additional performance metrics: (1) The average time for robots to reach the target area $\bar{t}_{dest}$. (2) The average DMPC calculation time per robot $\bar{t}_{cal}$.

A. Ackermann AGV Experiments

This section evaluates C-ROVER with ten Ackermann AGVs in an experimental field of size 13m×11m (Fig. 10). Ackermann AGVs are Wheeltec R550 ground robots, with a length of 0.29m and a width of 0.19m. C-ROVER is

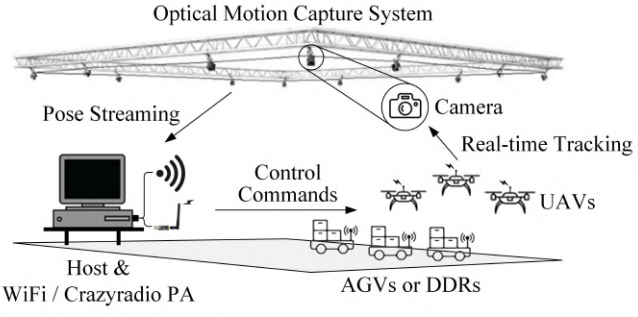

Fig. 9. Illustration of real-world experiment setup. An FZMotion/Optitrack optical motion capture system streams real-time robot poses to a host running C-ROVER, which continuously computes and dispatches commands over Wi-Fi (for AGVs and DDRs) or Crazyradio PA (for UAVs).

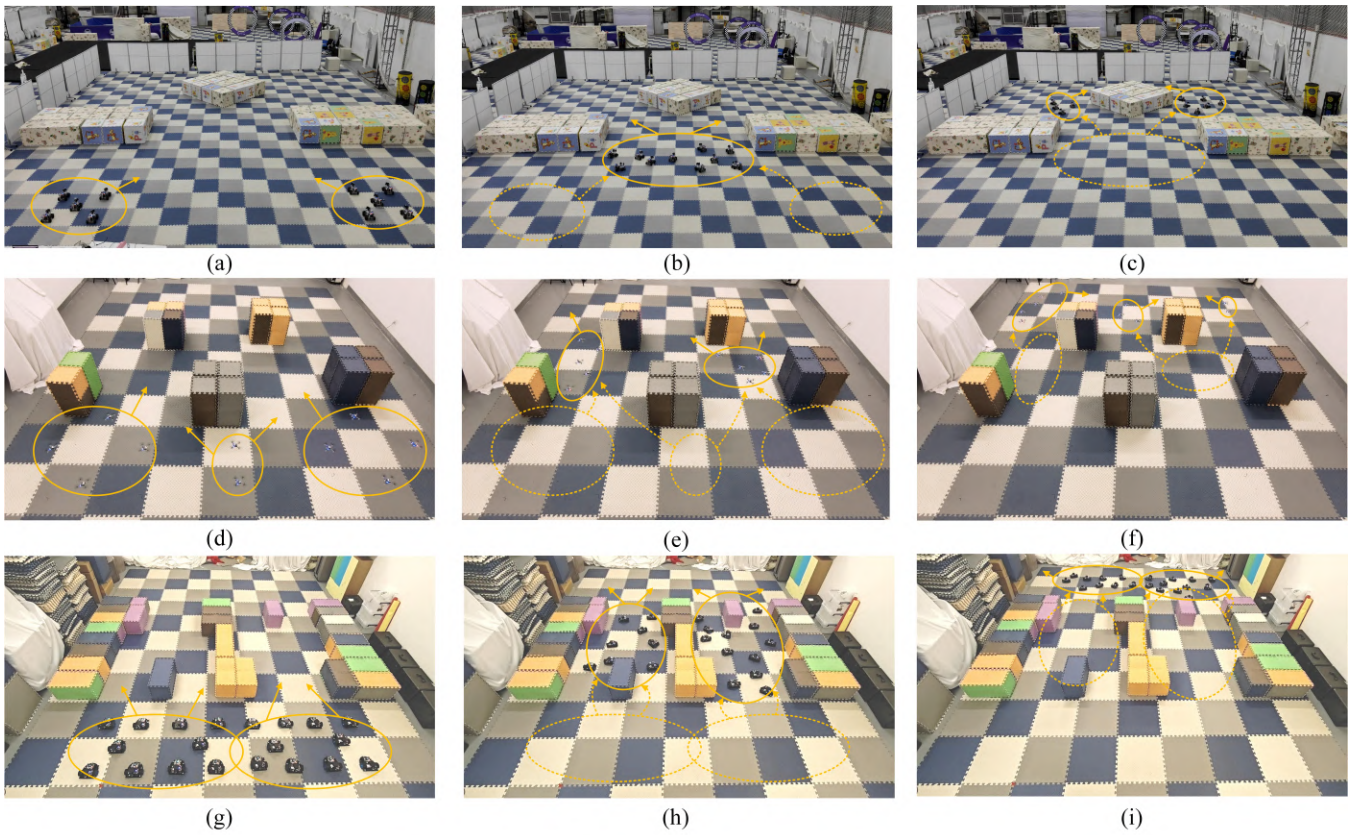

Fig. 10. The snapshots of AGV, UAV, and DDR experiments.

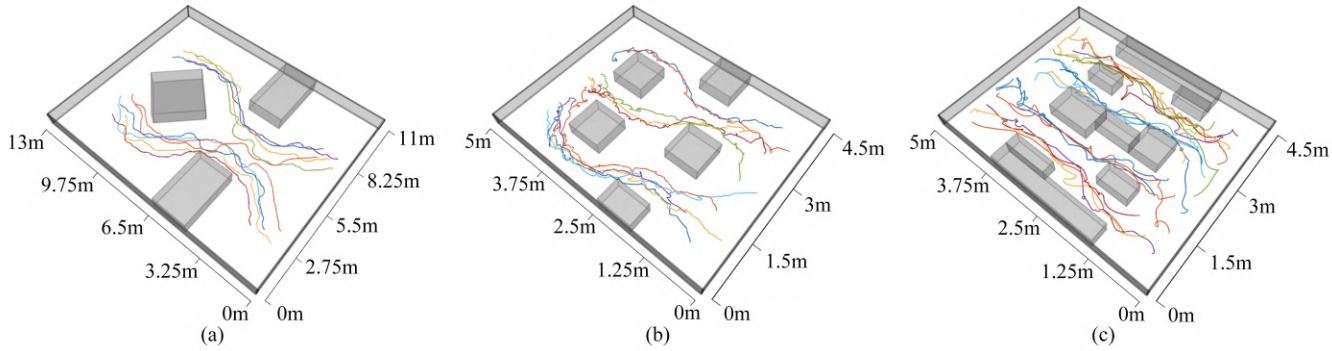

Fig. 11. The trajectories of AGVs, UAVs, and DDRs in real-world experiments. The left column of subfigures illustrates the robots' initial positions, and the middle and right columns of subfigures demonstrate instances of robots splitting and merging as they advance toward the target area.

TABLE V
THE PERFORMANCE OF REAL-WORLD EXPERIMENTS IN AGVS, UAVS, AND DDRS TASK SCENARIOS

Task Scenario	$\bar{t}_{dest}$ (s)	$\bar{t}_p$	$\bar{t}$	$\bar{t}_{cal}$ (s)	$\overline{D}$ (m)	$\overline{\min(d_{io})}$ (m)	$\overline{\min(d_{ij})}$ (m)
AGVs	151.243 (3.783)	84.357 (0.729)	241.66 (3.661)	0.009 (0.001)	12.429 (0.775)	0.356 (0.224)	0.256 (0.062)
UAVs	72.486 (2.741)	21.277 (0.400)	108.370 (1.885)	0.012 (0.0003)	6.355 (0.635)	0.154 (0.055)	0.154 (0.017)
DDRs	79.492 (0.252)	14.471 (1.293)	290.908 (2.289)	0.021 (0.0006)	5.279 (0.514)	0.125 (0.031)	0.107 (0.014)

implemented in Python 3.8 with multiprocessing, including one main process for planning GMM trajectories and ten subprocesses for robot control. The DMPC controller control frequency is 20Hz, and the maximum speed and angular velocity of AGV are set to be 0.1m/s and 45°/s, respectively.

Snapshots of the experiment are shown in Fig. 10, where the AGV swarm is navigating toward the target area. In Fig. 10(b), as all AGVs approach the passage, the limited

passage capacity causes the two groups to merge into one, after which the swarm reaches a bifurcation point and splits into two groups again, moving in two different directions, as illustrated in Fig. 10(c). Fig. 11(a) shows the complete trajectories of all AGVs, highlighting C-ROVER's flexibility for swarm splitting and merging. Tab. V presents the performance metrics, showing the mean and standard deviation (in parentheses) of three repeated trials, which demonstrates that C-ROVER ensures efficient and safe navigation for AGVs.

B. UAV Experiments

This section evaluates C-ROVER with ten Crazyflie 2.1 miniature quadcopters (92 × 92 × 29mm) in a 4.5m × 5m indoor environment. Each UAV utilizes an STM32F405 as its central processor for self-control and communication with the host computer via Crazyradio PA. The motion capture system sends all UAV position data to the Crazyswarm system installed on the host. The DMPC controller operates at 50Hz, with the maximum UAV speed set to be 0.3m/s, and the flight altitude at 0.6m.

We test the UAV swarm in a map, which contains small, scattered obstacles (Fig. 10(d)-(f)). The whole trajectory of the UAV swarm (Fig. 11(b)) shows that C-ROVER allows the UAV swarm to split and merge to avoid obstacles and pass through narrow passages due to the risk-aware design of the covariance matrix $\mathbf{\Sigma}_c^j$.

C. DDR Experiments

This section evaluates C-ROVER with twenty DDRs (radius: 0.1m) in a 4.5m × 5m indoor environment. Each robot is equipped with an ESP32 onboard processor for local control and communicates with a host computer via WLAN. The DMPC controller runs at 10Hz, with a maximum linear velocity of 0.15m/s and a maximum angular velocity limited to 90°/s.

The robot swarm is evaluated in a map featuring narrow passages and non-convex obstacles (Fig. 10(g)-(i)). As shown in Fig. 11(c), even with a larger number of robots, the excellent scalability of the proposed algorithm enables all robots to reach their destinations without collisions. Tab. V summarizes the mean and standard deviation (in parentheses) of three repeated trials, demonstrating safe and real-time navigation of C-ROVER.

IX. Conclusion

In this work, we present C-ROVER, a closed-loop motion planning and control framework for large-scale robotic swarms. By representing the swarm distribution with time-variant GMMs, C-ROVER enables flexible swarm behaviors such as splitting and merging to navigate around obstacles. To guide macroscopic swarm planning, we introduce a novel risk-aware planning space discretization, where the GCs are optimized to minimize CVaR-based collision risk while maximizing spatial coverage. The integration of stochastic SDF and CVaR provides a principled mechanism for quantifying collision risk, ensuring safe navigation in cluttered environments. Additionally, C-ROVER proposes a sequential optimization framework that supports efficient online planning, making it highly scalable to large swarm sizes. Extensive simulations and real-world experiments demonstrate the effectiveness of C-ROVER in terms of scalability, trajectory flexibility, and risk management. In future work, we plan to extend C-ROVER to unknown or dynamic environments, further enhancing its applicability to real-world scenarios.

References

[1] R. K. Ramachandran, N. Fronda, J. A. Preiss, Z. Dai, and G. S. Sukhatme, "Resilient multi-robot multi-target tracking," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 3, pp. 4311–4327, 2023.

[2] L. Chen, Y. Wang, Z. Miao, M. Feng, Y. Wang, Y. Mo, W. He, H. Wang, and D. Wang, "Str: Spatial-temporal retnet for distributed multi-robot navigation," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 10429–10441, 2025.

[3] A. Miele, M. Lippi, and A. Gasparri, "A distributed framework for integrated task allocation and safe coordination in networked multi-robot systems," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 11219–11238, 2025.

[4] P. Mao, R. Fu, and Q. Quan, "Optimal virtual tube planning and control for swarm robotics," *The International Journal of Robotics Research*, vol. 43, no. 5, pp. 602–627, 2024.

[5] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, *et al.*, "Swarm of micro flying robots in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm5954, 2022.

[6] P. Zhu, C. Liu, and S. Ferrari, "Adaptive online distributed optimal control of very-large-scale robotic systems," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 2, pp. 678–689, 2021.

[7] J. Yin, Z. Zhang, E. Theodorou, and P. Tsiotras, "Trajectory distribution control for model predictive path integral control using covariance steering," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 1478–1484, 2022.

[8] Y. Chen, "Density control of interacting agent systems," *IEEE Transactions on Automatic Control*, vol. 69, no. 1, pp. 246–260, 2024.

[9] K. Cui, M. Li, C. Fabian, and H. Koeppl, "Scalable task-driven robotic swarm control via collision avoidance and learning mean-field control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1192–1199, 2023.

[10] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1086–1095, 2019.

[11] E. Soria, F. Schiano, and D. Floreano, "Predictive control of aerial swarms in cluttered environments," *Nature Machine Intelligence*, vol. 3, no. 6, pp. 545–554, 2021.

[12] E. Soria, F. Schiano, and D. Floreano, "Distributed predictive drone swarms in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 73–80, 2021.

[13] A. Navsalkar and A. R. Hota, "Data-driven risk-sensitive model predictive control for safe navigation in multi-robot systems," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1442–1448, IEEE, 2023.

[14] S. Ma, M. Hou, X. Ye, and H. Zhou, "High-dimensional optimal density control with wasserstein metric matching," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 6813–6818, 2023.

[15] V. Krishnan and S. Martínez, "Distributed optimal transport for the deployment of swarms," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 4583–4588, 2018.

[16] S. Biswal, K. Elamvazhuthi, and S. Berman, "Decentralized control of multiagent systems using local density feedback," *IEEE Transactions on Automatic Control*, vol. 67, no. 8, pp. 3920–3932, 2021.

[17] K. Elamvazhuthi and S. Berman, "Mean-field models in swarm robotics: A survey," *Bioinspiration & Biomimetics*, vol. 15, no. 1, p. 015001, 2019.

[18] T. Zheng, Q. Han, and H. Lin, "Transporting robotic swarms via mean-field feedback control," *IEEE Transactions on Automatic Control*, vol. 67, no. 8, pp. 4170–4177, 2021.

[19] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017.

[20] X. Yang, Y. Hu, H. Gao, K. Ding, Z. Li, P. Zhu, Y. Sun, and C. Liu, "Risk-aware non-myopic motion planner for large-scale robotic swarm using cvar constraints," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5784–5790, IEEE, 2024.

[21] Y. Hu, X. Yang, K. Zhou, Q. Liu, K. Ding, H. Gao, P. Zhu, and C. Liu, "Swarmprm: Probabilistic roadmap motion planning for large-scale swarm robotic systems," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10222–10228, IEEE, 2024.

[22] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research: The 14th International Symposium ISRR*, pp. 3–19, Springer, 2011.

[23] S. Tang and V. Kumar, "Safe and complete trajectory generation for robot teams with higher-order dynamics," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1894–1901, IEEE, 2016.

[24] W. Hönig, J. A. Preiss, T. S. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 856–869, 2018.

[25] F. Meng, L. Chen, H. Ma, J. Wang, and M. Q.-H. Meng, "Learning-based risk-bounded path planning under environmental uncertainty," *IEEE Transactions on Automation Science and Engineering*, 2023.

[26] W. Han, A. Jasour, and B. Williams, "Non-gaussian risk bounded trajectory optimization for stochastic nonlinear systems in uncertain environments," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 11044–11050, IEEE, 2022.

[27] A. Jasour, W. Han, and B. C. Williams, "Convex risk-bounded continuous-time trajectory planning and tube design in uncertain nonconvex environments," *The International Journal of Robotics Research*, vol. 42, no. 10, pp. 705–728, 2023.

[28] R. T. Rockafellar, S. Uryasev, *et al.*, "Optimization of conditional value-at-risk," *Journal of Risk*, vol. 2, pp. 21–42, 2000.

[29] A. Hakobyan, G. C. Kim, and I. Yang, "Risk-aware motion planning and control using cvar-constrained optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3924–3931, 2019.

[30] D. D. Fan, K. Otsu, Y. Kubo, A. Dixit, J. Burdick, and A.-A. Agha-Mohammadi, "Step: Stochastic traversability evaluation and planning for risk-aware off-road navigation," in *Robotics: Science and Systems*, pp. 1–21, RSS Foundation, 2021.

[31] Y. Chen, T. T. Georgiou, and A. Tannenbaum, "Optimal transport for gaussian mixture models," *IEEE Access*, vol. 7, pp. 6269–6278, 2018.

[32] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 1988.

[33] G. Van Den Bergen, "Proximity queries and penetration depth computation on 3d game objects," in *Game Developers Conference*, vol. 170, p. 209, 2001.

[34] R. T. Rockafellar and S. Uryasev, "Conditional value-at-risk for general loss distributions," *Journal of Banking & Finance*, vol. 26, no. 7, pp. 1443–1471, 2002.

[35] M. Norton, V. Khokhlov, and S. Uryasev, "Calculating cvar and bpoe for common probability distributions with application to portfolio optimization and density estimation," *Annals of Operations Research*, vol. 299, pp. 1281–1315, 2021.

[36] K. N. Plataniotis and D. Hatzinakos, "Gaussian mixtures and their applications to signal processing," *Advanced Signal Processing Handbook*, pp. 89–124, 2017.

[37] G. Pertaia and S. Uryasev, "Fitting heavy-tailed mixture models with cvar constraints," *Dependence Modeling*, vol. 7, no. 1, pp. 365–374, 2019.

[38] D. W. Peterson, "A review of constraint qualifications in finite-dimensional spaces," *SIAM Review*, vol. 15, no. 3, pp. 639–654, 1973.

[39] F. Facchinei, L. Lampariello, and G. Scutari, "Feasible methods for nonconvex nonsmooth problems with applications in green communications," *Mathematical Programming*, vol. 164, no. 1-2, pp. 55–90, 2017.

[40] P. Mao, S. Lv, and Q. Quan, "Tube rrt*: Efficient homotopic path planning for swarm robotics passing-through large-scale obstacle environments," *IEEE Robotics and Automation Letters*, vol. 10, no. 3, pp. 2247–2254, 2025.

[41] X. Yang, H. Gao, P. Zhu, and C. Liu, "Risk-aware motion planning for very-large-scale robotics systems using conditional value-at-risk," in *International Conference on Intelligent Robotics and Applications (ICIRA)*, pp. 513–525, Springer, 2023.

[42] J. M. Danskin, *The theory of max-min and its application to weapons allocation problems*, vol. 5. Springer Science & Business Media, 2012.

APPENDIX I
PROOF OF LEMMA 1

Proof: We adopt the assumption that the closest point $\boldsymbol{o}_\iota^*$ and the normal vector $\boldsymbol{n}^j$ remain unchanged for any neighbouring samples of the mean position from $\boldsymbol{X}^j$, when calculating $sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$ [41]. Then $sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$ can be approximated with first-order Taylor expansion as follows,

$$sd(\boldsymbol{X}^j, \mathcal{O}_\iota) \approx sd(\boldsymbol{\mu}^j, \mathcal{O}_\iota) + \nabla sd(\boldsymbol{X}^j, \mathcal{O}_\iota)|_{\boldsymbol{X}^j=\boldsymbol{\mu}^j}(\boldsymbol{X}^j - \boldsymbol{\mu}^j), \tag{31}$$

where $\nabla sd(\boldsymbol{X}^j, \mathcal{O}_\iota)|_{\boldsymbol{X}^j=\boldsymbol{\mu}^j} = (\boldsymbol{n}^j)^T$. Equation (31) renders a linear transformation from the Gaussian r.v. $(\boldsymbol{X}^j - \boldsymbol{\mu}^j)$ to $sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$. Therefore, $sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$ also follows a Gaussian distribution with the mean $sd(\boldsymbol{\mu}^j, \mathcal{O}_\iota)$ and covariance $(\boldsymbol{n}^j)^T \boldsymbol{\Sigma}^j \boldsymbol{n}^j$.

APPENDIX II
PROOF OF LEMMA 2

Proof: To build the relationship between the distributions of r.v. $\sum_{j=1}^{N} \omega_j sd(\boldsymbol{X}^j, \mathcal{O}_\iota)$ and r.v. $sd(\boldsymbol{X}, \mathcal{O}_\iota)$, we derive the relationship between two probabilities $\sum_{j=1}^{N} \omega_j Pr(sd(\boldsymbol{X}^j, \mathcal{O}_\iota))$ and $Pr(sd(\boldsymbol{X}, \mathcal{O}_\iota))$. Here we have a signed distance mapping $sd : \mathcal{X} \subset \mathbb{R}^2 \to \mathcal{Y} \subset \mathbb{R}$. For notation simplexity, we omit the obstacle parameter in sd in the context of considering one static obstacle and define $Y^j = sd(\boldsymbol{X}^j, \mathcal{O}_\iota) = sd(\boldsymbol{X}^j)$ and $Y = sd(\boldsymbol{X}, \mathcal{O}_\iota) = sd(\boldsymbol{X})$. We then define the preimage of sd, i.e., $sd^{-1} : \mathcal{Y} \subset \mathbb{R} \to \mathcal{X} \subset \mathbb{R}^2$, such that $sd^{-1}(\mathcal{Y}) = \{\boldsymbol{X} \in \mathcal{X} | sd(\boldsymbol{X}) \in \mathcal{Y}\}$. As $\mathcal{Y}$ is a Borel set, the $sd^{-1}(\mathcal{Y})$ is also measurable. Due to the local monotonicity of $sd(\cdot)$, we denote subsets $\mathcal{Y}_q \subset \mathcal{Y}, \forall q \in Q$, on which the $sd(\cdot)$ is monotonic. The probability $Pr(Y \in \mathcal{Y})$ is derived as follow,

$$\begin{aligned}
Pr(Y \in \mathcal{Y}) &= Pr(sd(\boldsymbol{X}) \in \mathcal{Y}) && \text{(32a)}\\
&= \sum_{q=1}^{Q} Pr(sd(\boldsymbol{X}) \in \mathcal{Y}_q) && \text{(32b)}\\
&= \sum_{q=1}^{Q} Pr(\boldsymbol{X} \in sd^{-1}(\mathcal{Y}_q)) && \text{(32c)}\\
&= Pr(\boldsymbol{X} \in sd^{-1}(\mathcal{Y})) && \text{(32d)}\\
&= \int_{\mathcal{X}} p(\boldsymbol{X} = \boldsymbol{x}) d\boldsymbol{x} && \text{(32e)}\\
&= \int_{\mathcal{X}} \sum_{j=1}^{N} p(\boldsymbol{X} = \boldsymbol{x} | c(\boldsymbol{X}) = j) Pr(c(\boldsymbol{X}) = j) d\boldsymbol{x} && \text{(32f)}\\
&= \sum_{j=1}^{N} \int_{\mathcal{X}} p(\boldsymbol{X} = \boldsymbol{x} | c(\boldsymbol{X}) = j) \omega_j d\boldsymbol{x} && \text{(32g)}\\
&= \sum_{j=1}^{N} \omega_j \int_{\mathcal{X}} p(\boldsymbol{X} = \boldsymbol{x} | c(\boldsymbol{X}) = j) d\boldsymbol{x} && \text{(32h)}
\end{aligned}$$

$$= \sum_{j=1}^{N} \omega_j Pr(sd^{-1}(\mathcal{Y})|c(\boldsymbol{X}) = j) \tag{32i}$$

$$= \sum_{j=1}^{N} \omega_j Pr(Y^j \in \mathcal{Y}), \tag{32j}$$

where $p(\cdot)$ denotes a PDF. As there exists a finite number of extrema in $sd(\cdot)$, eq. (32a) to eq. (32d) can be established. The derivation to eq. (32e) holds because of the continuity of $sd(\cdot)$. Derivation from eq. (32e) to eq. (32f) is based on the definition of GMM and Bayes' theorem, with $c(\boldsymbol{X}) = j$ representing the event that $\boldsymbol{X}$ is sampled from the jth GC. The reasoning from eq. (32h) to eq. (32j) involves the SDF defined under Gaussian uncertainty.

APPENDIX III
PROOF OF THEOREM 1

Proof: According to the definition of CVaR in eq. (6) and the general properties of GMM, $CVaR_\alpha(Y)$ can be defined as

$$CVaR_\alpha(Y) = \min_{z\in\mathbb{R}} \mathbb{E}\left[z + \frac{[Y-z]^+}{\alpha}\right] \tag{33a}$$

$$= \min_{z\in\mathbb{R}} \left[z + \int_z^{+\infty} \frac{y-z}{\alpha} p(Y=y)dy\right] \tag{33b}$$

$$= \min_{z\in\mathbb{R}} \left[z + \int_z^{+\infty} \frac{y-z}{\alpha} \sum_{j=1}^{N} \omega_j p(Y^j=y)dy\right], \tag{33c}$$

For notation simplicity, we define

$$Q(\boldsymbol{\omega}, z) = z + \int_z^{+\infty} \frac{y-z}{\alpha} \sum_{j=1}^{N} \omega_j p(Y^j=y)dy, \tag{34}$$

where $\boldsymbol{\omega} = [\omega_1, \omega_2, \cdots, \omega_N]$, and thus we obtain

$$CVaR_\alpha(Y) = \min_{z\in\mathbb{R}} Q(\boldsymbol{\omega}, z). \tag{35}$$

We then take the partial derivative of $Q(\boldsymbol{\omega}, z)$ concerning z as

$$\frac{\partial Q(\boldsymbol{\omega}, z)}{\partial z} = 1 - \int_z^{\infty} \frac{p(Y=y)}{\alpha} dy. \tag{36}$$

As eq. (36) is monotonically increasing, we can obtain the unique minimizer of $Q(\boldsymbol{\omega}, z)$ as

$$z^* = \arg\min_{z\in\mathbb{R}} Q(\boldsymbol{\omega}, z) = VaR_\alpha(Y). \tag{37}$$

Plugging z^* into eq. (33c), the $CVaR_\alpha(Y)$ can be further derived as follows,

$$CVaR_\alpha(Y)$$

$$= VaR_\alpha(Y) + \int_{VaR_\alpha(Y)}^{+\infty} \frac{y - VaR_\alpha(Y)}{\alpha} \sum_{j=1}^{N} \omega_j p(Y^j=y)dy \tag{38a}$$

$$= VaR_\alpha(Y) + \frac{1}{\alpha} \sum_{j=1}^{N} \omega_j \left[\int_{VaR_\alpha(Y)}^{+\infty} yp(Y^j=y)dy - \alpha_j VaR_\alpha(Y)\right] \tag{38b}$$

$$= \frac{1}{\alpha} \sum_{j=1}^{N} \omega_j \int_{VaR_\alpha(Y)}^{+\infty} yp(Y^j=y)dy \tag{38c}$$

$$= \frac{1}{\alpha} \sum_{j=1}^{N} \omega_j \alpha_j CVaR_{\alpha_j}(Y^j). \tag{38d}$$

To obtain eqs. (38b) and (38c), we define an auxiliary variable α_j as

$$\alpha_j = \int_{VaR_\alpha(Y)}^{+\infty} p(Y^j=y)dy, \alpha = \sum_{j=1}^{N} \omega_j \alpha_j, \tag{39}$$

which refers to the tail probability of the jth Gaussian SDF distribution at the $VaR_\alpha(Y)$-th quantile. The transformation to eq. (38d) is based on the relationship of CVaR and VaR given in eq. (6).

APPENDIX IV
DERIVATION OF GRADIENT OF CVAR OF A GMM RANDOM VARIABLE

According to the Danskin's theorem [42], $CVaR_\alpha(Y)$ defined by eq. (35) is differentiable w.r.t. $\boldsymbol{\omega}$, since z^* is the unique minimizer of $Q(\boldsymbol{\omega}, z)$ as shown in eq. (37). And the partial derivatives can be computed by

$$\frac{\partial CVaR_\alpha(Y)}{\partial \omega_j} = \left.\frac{\partial Q(\boldsymbol{\omega}, z)}{\partial \omega_j}\right|_{z=VaR_\alpha(Y)} \tag{40a}$$

$$= \frac{1}{\alpha} \int_{VaR_\alpha(Y)}^{+\infty} yp(Y^j=y)dy - \frac{VaR_\alpha(Y)}{\alpha} \int_{VaR_\alpha(Y)}^{+\infty} p(Y^j=y)dy \tag{40b}$$

$$= \frac{\alpha_j}{\alpha} CVaR_{\alpha_j}(Y^j) - \frac{\alpha_j}{\alpha} VaR_\alpha(Y). \tag{40c}$$

The transformation from eq. (40b) to eq. (40c) leverages the relationship of CVaR and VaR given in eq. (6).

APPENDIX V
PROOF OF PROPOSITION 2

Proof: It is evident that the equality in eq. (29) holds only when $N = 1$. For the general case of a GMM distribution, we consider $N > 1$. To establish the inequality $VaR_\alpha(Y) < \max_{j=1:N} VaR_\alpha(Y^j)$, we begin by employing a proof by contradiction. If $VaR_\alpha(Y) \geqslant \max_{j=1:N} VaR_\alpha(Y^j)$, the probability of the GMM that exceeds $VaR_\alpha(Y)$ quantile can be denoted as

$$\alpha_1\omega_1 + \alpha_2\omega_2 + \cdots + \alpha_N\omega_N < \sum_{j=1}^{N} \omega_j \alpha = \alpha, \tag{41}$$

where $\alpha_1, \alpha_2, \cdots, \alpha_N$ denote the tail probability of each GC at the $VaR_\alpha(Y)$-th quantile, because α decreases monotonically w.r.t. VaR. This contradicts the definition of $VaR_\alpha(Y)$.

Thus the inequality $VaR_\alpha(Y) < \max_{j=1:N} VaR_\alpha(Y^j)$ holds.

Similarly, we can prove the inequality for the other side.