SwarmPRM: Probabilistic Roadmap Motion Planning for Large-Scale Swarm Robotic Systems

Yunze Hu[1], Xuru Yang[1], Kangjie Zhou[1], Qinghang Liu[1], Kang Ding[1], Han Gao[1], Pingping Zhu[2], and Chang Liu[1]

***Abstract*— Large-scale swarm robotic systems consisting of numerous cooperative agents show considerable promise for performing autonomous tasks across various sectors. Nonetheless, traditional motion planning approaches often face a trade-off between scalability and solution quality due to the exponential growth of the joint state space of robots. In response, this work proposes SwarmPRM, a hierarchical, scalable, computationally efficient, and risk-aware sampling-based motion planning approach for large-scale swarm robots. SwarmPRM utilizes a Gaussian Mixture Model (GMM) to represent the swarm's macroscopic state and constructs a Probabilistic Roadmap in Gaussian space, referred to as the Gaussian roadmap, to generate a transport trajectory of GMM. This trajectory is then followed by each robot at the microscopic stage. To enhance trajectory safety, SwarmPRM incorporates the conditional value-at-risk (CVaR) in the collision checking process to impart the property of risk awareness to the constructed Gaussian roadmap. SwarmPRM then crafts a linear programming formulation to compute the optimal GMM transport trajectory within this roadmap. Extensive simulations demonstrate that SwarmPRM outperforms state-of-the-art methods in computational efficiency, scalability, and trajectory quality while offering the capability to adjust the risk tolerance of generated trajectories.**

I. Introduction

Large-scale swarm robotic systems comprised of numerous autonomous robots hold great promise for executing diverse tasks such as surveillance [1], environmental exploration [2], and search and rescue [3]. In recent years, there has been a surge in interest towards developing motion planning techniques for large-scale swarm robots [4–6].

Sampling-based algorithms [7] as a prominent motion planning technique have demonstrated significant potential for application in swarm robotic systems. For example, Čáp et al. [8] adapted RRT* to solve the multi-agent motion planning problem in a centralized manner by treating the union of all agents as a single system. However, this approach is computationally intractable due to the exponential growth of the search space as the swarm size increases. Hönig et al. [4] enhanced scalability by first generating a sparse probabilistic roadmap (PRM) and then utilizing multi-agent path finding algorithms for path planning subject to inter-robot constraints in the space discretized by the roadmap. Yet, the sparsity of the roadmap compromises the trajectory cost in favor of computational efficiency. Shome et al. [9] proposed dRRT*, an informed, asymptotically optimal sampling-based motion planning algorithm that includes constructing PRMs for individual robots and building a search tree without requiring an explicit representation of the joint motion graph. Sampling-based approaches have proven effective for trajectory planning, particularly in cluttered environments. Nevertheless, these methods inevitably face a trade-off between scalability and the cost of the trajectory due to the exponential growth of the joint state space of all robots. Besides, as the swarm size increases, ensuring inter-robot constraint satisfaction becomes increasingly difficult.

Recently, the hierarchical strategy has emerged as a promising approach for the motion planning of large-scale swarm robots. The hierarchical strategy consists of macro-

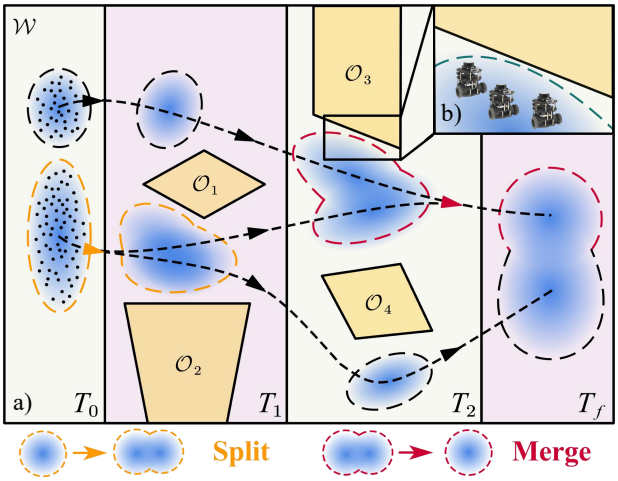

Fig. 1: **Illustration of the hierarchical motion planning for a large-scale robot swarm.** (a) The macroscopic planning of the swarm robots. In the two-dimensional cluttered environment $\mathcal{W}$ with four obstacles $\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3, \mathcal{O}_4$, the swarm's macroscopic state, represented as a PDF, is depicted as a blue cloud, and individual robots at time T_0 are represented by black dots. The macroscopic trajectory, as represented by the black dotted lines and colored arrows, guides the swarm from initial area at time T_0 to the target area at time T_f, passing through two intermediate macroscopic states at time T_1 and T_2, respectively. The trajectory may involve "splitting" or "merging" maneuvers, depicted by the orange and red dotted lines and arrows, respectively. (b) The microscopic state of robots in the swarm. The robots track the macroscopic state while avoiding collision with obstacles.

[1]Yunze Hu, Xuru Yang, Kangjie Zhou, Qinghang Liu, Kang Ding, Han Gao, and Chang Liu are with the Department of Advanced Manufacturing and Robotics, College of Engineering, Peking University, Beijing 100871, China (hu_yun_ze@stu.pku.edu.cn; xuru.yang@stu.pku.edu.cn; kangjiezhou@pku.edu.cn; californium@stu.pku.edu.cn; kangding@stu.pku.edu.cn; hangaocoe@pku.edu.cn; changliucoe@pku.edu.cn). All correspondences should be sent to Chang Liu.

[2]Pingping Zhu is with the Department of Computer Sciences and Electrical Engineering (CSEE), Marshall University, Huntington, WV 25755, USA (zhup@marshall.edu).

scopic and microscopic planning stages. At the macroscopic stage, the robot swarm is treated as an entirety, and the macroscopic swarm state is usually represented as a probability density function (PDF). At the microscopic stage, robots autonomously match the PDF by coordinating the distribution of their positions. The main benefit of employing a hierarchical strategy is that the computational complexity of macroscopic planning becomes independent of the size of the swarm. This is made possible by abstracting the detailed behaviors and positions of individual agents into a collective "swarm state", allowing for scalable and efficient management regardless of the swarm's size. Adopting this strategy, Rudd et al. [10] employed parametrized PDFs to represent the macroscopic states of the swarm and calculated the optimal robot distribution by solving a non-convex, constrained trajectory optimization problem. The computational burden is very high, though, hindering its real-time implementation in practice. Zhu et al. [5] proposed an ADOC approach that modeled the macroscopic state of the swarm as a Gaussian mixture model (GMM), and utilized the optimal mass transport (OMT) theory to plan the PDF trajectory. ADOC shows desirable real-time performance. However, using a predetermined set of Gaussian collocation points for GMM trajectory generation limits the flexibility and adaptability to complex spatial challenges in trajectory planning, thus often yielding suboptimal, costly trajectories in cluttered environments.

In this work, we combine the advantages of both sampling-based approaches and hierarchical strategy to propose a sampling-based, hierarchical motion planning approach for robotic swarms, namely SwarmPRM. At the macroscopic stage, the collective swarm is presented by a GMM, and a PRM is constructed to generate GMM trajectories. At the microscopic stage, individual robots track the PDF via a computationally efficient local tracking control law.

A significant challenge in SwarmPRM involves collision checking during the sampling process, as the samples are PDFs rather than deterministic states typical of traditional sampling-based methods. Consequently, there arises a necessity for a risk measure to assess the likelihood of collisions between a PDF sample and obstalces. To overcome this difficulty, we propose to use the conditional value-at-risk (CVaR) as the risk measure for collision checking in the sampling process. CVaR is a coherent measure [11] that quantifies the potential losses beyond a certain confidence level. Since CVaR can take into account the tail distribution through conditional expectation, which enables the discerning of rare events, it recently gained significant attention in the robotics community [12, 13].

In general, SwarmPRM is a novel sampling-based, hierarchical approach for the motion planning of large-scale swarm robotic systems, characterized by scalability, computational efficiency, risk awareness, and trajectory flexibility. The main contributions of this work are threefold:

1) We propose to sample in Gaussian parametric space to construct a Gaussian PRM where each node represents a distinct Gaussian distribution. The cost and geodesic path between nodes are determined under the Wasserstein metric.
2) We develop a systematic collision checking approach using CVaR as the risk measure when constructing the Gaussian PRM. Specifically, we compute the distribution of the signed distance function (SDF) between the samples and the obstacles, and constrain the CVaR of the SDF within the safety threshold.
3) We design a linear programming formulation to compute the optimal macroscopic GMM trajectory on the constructed roadmap. Through extensive simulations, we demonstrate the superiority of SwarmPRM over several state-of-the-art benchmark methods in aspects of computational efficiency, scalability, and trajectory quality. Furthermore, SwarmPRM exhibits risk awareness where the risk tolerance level can be easily adjusted, offering great flexibility in designing swarm behaviors, especially in cluttered environments.

II. Problem Formulation and Background

A. Problem Formulation

Consider a swarm robotic system represented by the set $\mathcal{R} = \{1, 2, \cdots, N\}$, where N denotes the number of homogeneous robots within a large two-dimensional cluttered environment $\mathcal{W} \subset \mathbb{R}^2$ containing N_{Obs} convex obstacles[1] $\boldsymbol{\mathcal{O}} = \{\mathcal{O}_1, \cdots, \mathcal{O}_{N_{Obs}}\}$, where $\mathcal{O}_i \subset \mathcal{W}, i = 1, \cdots, N_{Obs}$. The objective is to devise trajectories for all robots, directing them from initial positions to target areas. The obstacles are static and known a priori, and under the assumptions of connectivity and information sharing, the states of the robots are fully observable. The set of all robots' initial positions is denoted as $\mathcal{Q} = \{\boldsymbol{q}_1, \cdots, \boldsymbol{q}_N\}$, where $\boldsymbol{q_i} \in \mathcal{W}, i \in \mathcal{R}$.

We adopt a hierarchical motion planning strategy for the swarm robots (Fig. 1). At the macroscopic stage, we represent the entirety of the swarm as a PDF $\chi(t) \in \mathcal{P}(\mathcal{W})$ which is time-varying and constitutes a PDF trajectory with respect to time t. Here $\mathcal{P}(\mathcal{W})$ represents the space of PDFs with support $\mathcal{W}$. We model the initial and target distribution of swarm robots as two PDFs ρ_{T_0}, ρ_{T_f}, and aim at devising a swarm's macrsocopic state trajectory transitioning from ρ_{T_0} to ρ_{T_f} while simultaneously avoiding obstacles. At the microscopic stage, each robot autonomously tracks the PDF $\chi(t)$ while ensuring collision avoidance with obstacles and other robots.

At the macroscopic stage, the optimal time-varying PDF $\chi(t)$ can be calculated by solving the following optimization problem

$$\min_{\chi} \quad J(\chi(t)) \tag{1}$$

$$\text{s.t.} \quad \rho_{T_0} = \chi(T_0), \tag{1a}$$

$$\rho_{T_f} = \chi(T_f), \tag{1b}$$

$$R_\alpha(\chi(t)) \leqslant \delta, \forall t \in [T_0, T_f], \tag{1c}$$

[1]Note that our motion planning method is readily adaptable to environments containing non-convex obstacles via decomposing non-convex obstacles into the union of convex ones.

where (1a), (1b) denote the swarm's initial and target macroscopic PDF state constraints, respectively, and (1c) represents the collision avoidance constraint with R and δ denoting the risk measure and safe region threshold, respectively. The objective is to minimize the swarm's transport cost J, while adhering to the aforementioned constraints.

B. Sampling-Based Motion Planning Algorithms

Sampling-based Algorithms (SBAs) have been a prevalent strategy for motion planning in robotics [7, 14, 15]. SBAs are characterized by a triplet $(\mathcal{X}_{free}, \boldsymbol{x}_{init}, \boldsymbol{x}_{goal})$, where $\mathcal{X}_{free}$, $\boldsymbol{x}_{init}$, and $\boldsymbol{x}_{goal}$ represent the collision-free configuration space, the initial configuration, and the goal configuration, respectively. The objective is to find a collision-free trajectory $\tau : [0,1] \rightarrow \mathcal{X}_{free}$, satisfying $\tau(0) = \boldsymbol{x}_{init}$ and $\tau(1) = \boldsymbol{x}_{goal}$. SBAs operate by sampling the configuration space and constructing a tree or graph, which expresses the connectivity and the feasible paths within the space. In this work, we specifically consider PRM [7]. Initially, new samples are drawn from $\mathcal{X}_{free}$ to create graph nodes in the roadmap. Subsequently, each node undergoes a query process to identify neighboring nodes to which the transition is collision-free with respect to obstacles, and edges connecting the respective nodes are added to the roadmap. Finally, a graph search is performed to find a shortest path connecting $\boldsymbol{x}_{init}$ and $\boldsymbol{x}_{goal}$ in the constructed roadmap.

C. Optimal Transport Theory and Wasserstein Metric

The OMT theory [16] tackles the task of transporting masses from an initial distribution to a terminal one while maintaining mass continuity and minimizing associated costs. The Wasserstein distance W_2 is an important metric representing the minimal transport cost within the space of PDFs. The Wasserstein metric between two Gaussian distributions $g_1 = \mathcal{N}(\boldsymbol{\mu_1}, \Sigma_1), g_2 = \mathcal{N}(\boldsymbol{\mu_2}, \Sigma_2)$ is

$$W_2(g_1, g_2) = \Big\{ \|\boldsymbol{\mu_1} - \boldsymbol{\mu_2}\|^2 + tr\left[\Sigma_1 + \Sigma_2 - 2(\Sigma_1^{\frac{1}{2}}\Sigma_2\Sigma_1^{\frac{1}{2}})^{\frac{1}{2}}\right] \Big\}^{\frac{1}{2}}, \tag{2}$$

where $\|\cdot\|$ denotes the Euclidean distance and $tr(\cdot)$ represents the trace of a matrix. The geodesic path $\hat{g}_{1,2}(t), t \in [0,1]$ between g_1, g_2 is Gaussian with following mean and covariance [16]

$$\hat{\boldsymbol{\mu}}(t) = (1-t)\boldsymbol{\mu}_1 + t\boldsymbol{\mu}_2, \tag{3}$$

$$\hat{\Sigma}(t) = \Sigma_1^{-\frac{1}{2}}\big[(1-t)\Sigma_1 + t(\Sigma_1^{\frac{1}{2}}\Sigma_2\Sigma_1^{\frac{1}{2}})^{\frac{1}{2}}\big]^2\Sigma_1^{-\frac{1}{2}}. \tag{4}$$

There is no analytic expression for the Wasserstein metric between two GMMs. In response, a metric in the space of all GMMs, $\mathcal{GM}$, defined as

$$D(\varrho_1, \varrho_2) = \left\{ \min_{\pi \in \Pi(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)} \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} \Big[W_2(g_1^i, g_2^j)\Big]^2 \pi(i,j) \right\}^{\frac{1}{2}}, \tag{5}$$

has been proposed in [16] as an efficient approximation to the Wasserstein metric for two arbitrary GMMs $\varrho_1 = \sum_{i=1}^{N_1}\omega_1^i g_1^i, \varrho_2 = \sum_{j=1}^{N_2}\omega_2^j g_2^j \in \mathcal{GM}$, where $\boldsymbol{\omega}_1 = [\omega_1^1, \cdots, \omega_1^{N_1}]$ and $\boldsymbol{\omega}_2 = [\omega_2^1, \cdots, \omega_2^{N_2}]$ are associated weight vectors satisfying $\sum_{i=1}^{N_1}\omega_1^i = 1, \sum_{j=1}^{N_2}\omega_2^j = 1$, and $\Pi(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$ denotes the space of joint probability distributions between $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$. The geodesic path connecting ϱ_1 and ϱ_2 is given by

$$\hat{\varrho}_{1,2}(t) = \sum_{i,j} \pi^*(i,j)\hat{g}_{1,2}^{i,j}(t), t \in [0,1], \tag{6}$$

where $\pi^*(i,j)$ denotes the optimal joint distribution, and $\hat{g}_{1,2}^{i,j}(t)$ represents the geodesic path between Gaussian distributions g_1^i and g_2^j, which can be calculated based on Eqs. (3) and (4).

D. Conditional Value-at-Risk

Given a PDF and a risk tolerance level α, the CVaR computes the conditional expectation of the loss within the α worst-case quantile [12]. The CVaR of a Gaussian random variable $\upsilon \sim \mathcal{N}(\mu, \sigma^2)$ is

$$\text{CVaR}_\alpha(\upsilon) = \mu + \frac{\phi(\Phi^{-1}(1-\alpha))}{\alpha}\sigma, \tag{7}$$

where ϕ and Φ are the PDF and cumulative density function (CDF) of the standard normal distribution, respectively.

III. Constructing Risk-Aware Gaussian Roadmap

The swarm's macroscopic state is represented by a time-varying PDF $\chi(t), t \in [T_0, T_f]$. Because of the universal approximation property of GMMs [17], without loss of generality, we approximate $\chi(t)$ as a GMM at each time instance. The swarm's initial and target macroscopic states are modeled as two GMMs ϱ_{T_0}, ϱ_{T_f}, and the optimal PDF transport trajectory is devised in GMM space $\mathcal{GM}$, i.e., $\forall t \in [T_0, T_f], \chi(t) \in \mathcal{GM}, \chi(T_0) = \varrho_{T_0}, \chi(T_f) = \varrho_{T_f}$. The distance metric Eq. (5) and the corresponding geodesic path Eq. (6) in GMM space suggest that the optimal transport between two GMMs can be achieved through transport weight allocation between their respective Gaussian components. Furthermore, the geodesic path Eq. (3) and Eq. (4) between two Gaussian distributions implies that each intermediate state along the optimal transport trajectory between two Gaussian distributions retains a Gaussian distribution. Therefore, the optimal GMM trajectory $\chi(t)$ can be achieved through a set of Gaussian distribution trajectories originating from the Gaussian components of ϱ_{T_0} and advancing towards those of ϱ_{T_f}.

To compute such trajectories, we propose to construct a risk-aware Gaussian roadmap at the macroscopic planning stage. In contrast to conventional roadmap construction methods [7], our approach samples each node as a Gaussian distribution, and determines the distance and geodesic path between each pair of nodes based on the Wasserstein metric. Furthermore, the risk measure CVaR is utilized to perform collision checking for each Gaussian node. This Gaussian roadmap is then used to generate GMM trajectory $\chi(t)$ between ϱ_{T_0} and ϱ_{T_f}, which will be detailed in Sec. IV.

A. Roadmap Construction

Denote $\mathcal{G}$ as the space of two-dimensional Gaussian distributions, where each element is represented as $g = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, with the mean $\boldsymbol{\mu} \in \mathcal{W}$, and covariance matrix

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}. \tag{8}$$

Here $\sigma_1^2, \sigma_2^2, \rho$ denote two variances and the correlation coefficient, respectively. The distance in $\mathcal{G}$ is defined based on the Wasserstein metric $W_2 : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$, and the cost function is the path length in this Gaussian space. The roadmap construction approach (Alg. 1) is outlined in the ROADMAPCONSTRUCTION function, which takes the number of samples n, the connection radius r, the set of obstacles $\boldsymbol{\mathcal{O}}$, and a set $\mathcal{D}$ representing the set of initial and target Gaussian distributions as inputs. Initially, a node set V is constructed through the combination of the set $\mathcal{D}$ as well as $n \in \mathbb{N}$ nodes generated by the function SAMPLEFREE (Line 1). Subsequently, for each node $g \in V$, the function NEIGHBOUR calculates the set V_{near} containing all nodes located in the neighbourhood of node g (Line 3). For each node $g' \in V_{near}$, the edges (g, g') and (g', g) are added to the edge set E, if the geodesic path $\hat{g}(t)$ defined based on the Wasserstein metric is a subset of the obstacle-free region in $\mathcal{G}$, which is checked by the function COLLISIONFREE (Line 4-7). Finally, a graph $G = (V, E)$ is constructed (Line 11). The functions SAMPLEFREE and NEIGHBOUR are detailed as follows, and the functions INFREE and COLLISIONFREE will be detailed in Sec. III-B.

1) SAMPLEFREE*:* The function SAMPLEFREE returns a set of $n \in \mathbb{N}$ nodes in free space. We utilize five-dimensional parameter vectors $\boldsymbol{v} = [x, y, \sigma_1, \sigma_2, \rho] \in \mathbb{R}^5$ to represent Gaussian distributions in $\mathcal{G}$, with mean $[x, y]$ and covariance matrix in the form of (8). Whenever the number of nodes in the set V is less than n, denoted as $|V| < n$, a node g is generated through sampling a parameter vector $\boldsymbol{v}$ (Line 13-14). Various sampling strategies, including random sampling and deterministic sampling methods [15], can be utilized to sample $\boldsymbol{v}$. If the Gaussian distribution g is in free space, as verified by the function INFREE, then it is added to the set V (Line 15-17).

2) NEIGHBOUR*:* The function NEIGHBOUR takes the node g, connection radius r, and the node set V' as inputs, and returns the set $V_{near} \subset V'$ containing all nodes located in the neighbourhood of node g. For each node g' in the set V', if $W_2(g, g')$ is no more than the connection radius r, then g' is added to the set V_{near} (Line 22-24).

B. Collision Checking Based on CVaR

We utilize the risk measure CVaR to perform collision checking between each Gaussian node and the obstacles $\boldsymbol{\mathcal{O}}$ during the roadmap construction procedure. We first derive via the linearization technique that PDF of the SDF between a Gaussian node and an obstacle can be approximated as a Gaussian distribution. Subsequently, we utilize CVaR to constrain the conditional expectation of the SDF within the α worst-case quantile to reside within the safe region, where α denotes the risk tolerance level.

Algorithm 1: Gaussian Roadmap Construction

Procedure ROADMAPCONSTRUCTION$(n, r, \boldsymbol{\mathcal{O}}, \mathcal{D})$
1: $V \leftarrow \mathcal{D} \cup$ SAMPLEFREE$(n, \boldsymbol{\mathcal{O}})$; $E \leftarrow \varnothing$
2: **for** $g \in V$ **do**
3: $V_{near} \leftarrow$ NEIGHBOUR$(V \backslash \{g\}, g, r)$
4: **for** $g' \in V_{near}$ **do**
5: Generate geodesic path $\hat{g}(t)$ from g and g' using (3) and (4)
6: **if** COLLISIONFREE$(\hat{g}(t), \boldsymbol{\mathcal{O}})$ **then**
7: $E \leftarrow E \cup \{(g, g')\} \cup \{(g', g)\}$
8: **end if**
9: **end for**
10: **end for**
11: **return** $G = (V, E)$

Procedure SAMPLEFREE$(n, \boldsymbol{\mathcal{O}})$
12: $V \leftarrow \varnothing$
13: **while** $|V| < n$ **do**
14: Generate Gaussian distribution $g \in \mathcal{G}$ through sampling parameter vector $\boldsymbol{v}$
15: **if** INFREE$(g, \boldsymbol{\mathcal{O}})$ **then**
16: $V \leftarrow V \cup \{g\}$
17: **end if**
18: **end while**
19: **return** V

Procedure NEIGHBOUR(V', g, r)
20: $V_{near} \leftarrow \varnothing$
21: **for** $g' \in V'$ **do**
22: **if** $W_2(g, g') \leqslant r$ **then**
23: $V_{near} \leftarrow V_{near} \cup \{g'\}$
24: **end if**
25: **end for**
26: **return** V_{near}

Procedure INFREE$(g, \boldsymbol{\mathcal{O}})$
27: **for all** $\mathcal{O} \in \boldsymbol{\mathcal{O}}$ **do**
28: Compute the negation of SDF using (11)
29: **if** (12) not holds **then**
30: **return false**
31: **end if**
32: **end for**
33: **return true**

1) SDF Linearization: The SDF quantifies the distance between two sets $\mathcal{A}$ and $\mathcal{B}$, defined as

$$s(\mathcal{A}, \mathcal{B}) = \begin{cases} \inf\{\|\boldsymbol{t}\|_2 \mid \boldsymbol{t} + \mathcal{A} \cap \mathcal{B} \neq \varnothing\}, \text{if} \mathcal{A} \cap \mathcal{B} = \varnothing, \\ -\inf\{\|\boldsymbol{t}\|_2 \mid \boldsymbol{t} + \mathcal{A} \cap \mathcal{B} = \varnothing\}, \text{if} \mathcal{A} \cap \mathcal{B} \neq \varnothing, \end{cases} \tag{9}$$

For a deterministic point $\boldsymbol{p}$ and a convex obstacle $\mathcal{O}$, we can obtain the signed distance $\hat{d} \in \mathbb{R}$, the closest points from the obstacle $\boldsymbol{p}_{\mathcal{O}} \in \mathcal{O}$, and the contact normal $\boldsymbol{n}(\boldsymbol{p}, \mathcal{O}) = sgn(\hat{d})(\boldsymbol{p}_{\mathcal{O}} - \boldsymbol{p})/\|\boldsymbol{p}_{\mathcal{O}} - \boldsymbol{p}\|$ utilizing Gilbert–Johnson–Keerthi (GJK) algorithm for non-intersecting sets [18] and Expanding Polytope Algorithm (EPA) for overlapping shapes [19].

For a stochastic point $\boldsymbol{p} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ and a deterministic obstacle $\mathcal{O}$, the SDF can be linearized as

$$s(\boldsymbol{p}, \mathcal{O}) \approx s(\boldsymbol{\mu}, \mathcal{O}) + \nabla s(\boldsymbol{p}, \mathcal{O}) \mid_{\boldsymbol{p}=\boldsymbol{\mu}} \cdot (\boldsymbol{p} - \boldsymbol{\mu}), \tag{10}$$

which approximates the obstacle as a half plane with outer

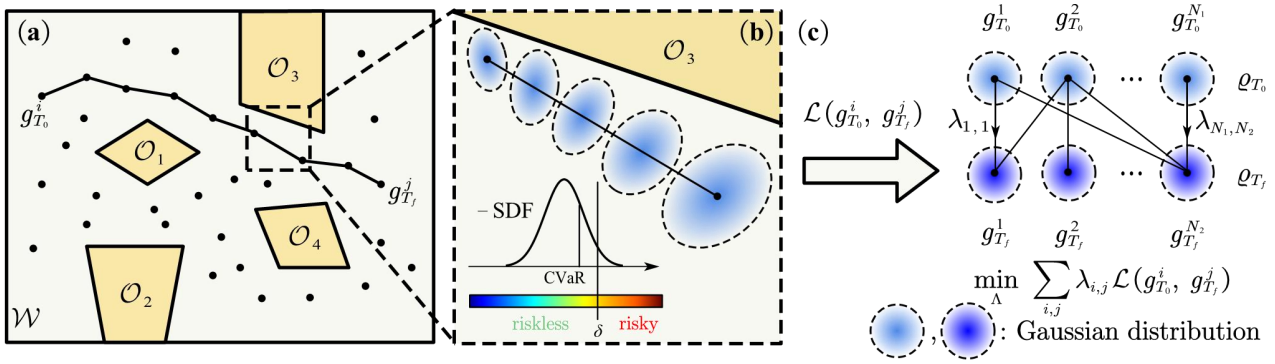

Fig. 2: **SwarmPRM motion planning process at the macroscopic stage.** (a) Projection of the risk-aware Gaussian roadmap onto the workspace $\mathcal{W}$, with the obstacles $\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3, \mathcal{O}_4$ colored in light orange. Each black dot represents a node in the Gaussian roadmap, which represents a two-dimensional Gaussian distribution. The black polyline depicts the shortest path between Gaussian distributions $g^i_{T_0}$ and $g^j_{T_f}$ on the roadmap. (b) Each segment of the polyline represents an edge in the Gaussian roadmap illustrating the shortest geodesic path between two nodes defined based on the Wasserstein metric. The CVaR associated with each node in the Gaussian roadmap is required to be below the safe region threshold δ. (c) Through graph search on the Gaussian roadmap, the lowest transport cost $\mathcal{L}(g^i_{T_0}, g^j_{T_f})$ between each pair of Gaussian distributions $(g^i_{T_0}, g^j_{T_f})$ can be computed. The optimal GMM trajectory on the Gaussian roadmap is then calculated by solving a linear programming problem.

normal vector $-\boldsymbol{n}(\boldsymbol{\mu}, \mathcal{O})$ so that $\nabla s(\boldsymbol{p}, \mathcal{O}) \mid_{\boldsymbol{p}=\boldsymbol{\mu}} = \boldsymbol{n}(\boldsymbol{\mu}, \mathcal{O})$. It is obvious that the random variable $s(\boldsymbol{p}, \mathcal{O})$ satisfies a Gaussian distribution with mean $s(\boldsymbol{\mu}, \mathcal{O})$ and covariance $\boldsymbol{n}^T \Sigma \boldsymbol{n}$ after linearization [13, 20], where $\boldsymbol{n} \triangleq \boldsymbol{n}(\boldsymbol{\mu}, \mathcal{O})$.

2) CVaR Collision Checking for Swarm Robots: Given a team of robots whose macroscopic state is represented by a random variable satisfying a Gaussian distribution $\boldsymbol{p} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ and an obstacle $\mathcal{O}$, the probabilistic distribution of the SDF $s(\boldsymbol{p}, \mathcal{O})$ can be approximated as a Gaussian distribution. Define η as the negation of the SDF,

$$\eta \triangleq -s(\boldsymbol{p}, \mathcal{O}) \approx \eta' \sim \mathcal{N}(-s(\boldsymbol{\mu}, \mathcal{O}), \boldsymbol{n}^T \Sigma \boldsymbol{n}), \tag{11}$$

where a greater value of η indicates a smaller SDF value between the robot team and the obstacle $\mathcal{O}$, and η is approximated by the random variable η' through (10). We subsequently compute the CVaR of η', and the collision avoidance requirement can be written as

$$\text{CVaR}_\alpha(\eta, \mathcal{O}) \approx \text{CVaR}_\alpha(\eta', \mathcal{O}) \leqslant \delta, \tag{12}$$

where α denotes the risk tolerance level, and $\delta \leqslant 0$ represents the safe region threshold, constraining the conditional expectation of opposite SDF within the α worst-case quantile to be no more than δ.

During the construction of the Gaussian roadmap, the function INFREE serves to check whether the nodes are collision-free with respect to obstacles, i.e., whether (12) holds for each environmental obstacle $\mathcal{O} \in \boldsymbol{\mathcal{O}}$, as presented in Alg. 1. For a Gaussian distribution g, all obstacles are queried to calculate the negation of the SDF which is a random variable following a Gaussian distribution (Line 27-28). Subsequently, the CVaR is calculated and the function INFREE returns **False** if (12) is not satisfied for any obstacle $\mathcal{O}$ (Line 29-30). Otherwise, the function INFREE returns **True** (Line 33). The function COLLISIONFREE (Line 6) checks whether the transition $\hat{g}(t)$ between two Gaussian distributions g and g' is collision-free with respect to obstacles $\boldsymbol{\mathcal{O}}$, i.e., $\forall g \in \hat{g}(t), t \in [0, 1]$, INFREE$(g, \boldsymbol{\mathcal{O}})$ returns **True**. In practical implementation, the collision checking for $\hat{g}(t)$ can be achieved approximately by performing INFREE assessments for a collection of Gaussian distributions interpolated between g and g' based on Eq. (3) and Eq. (4).

IV. SWARMPRM APPROACH FOR HIERARCHICAL MOTION PLANNING

The SwarmPRM approach (Alg. 2) consists of following two planning stages. At the macroscopic stage (Line 1-9), SwarmPRM leverages the risk-aware Gaussian roadmap constructed in Sec. III to address the optimization problem (1) within the GMM space and generate GMM trajectory $\chi(t)$. Subsequently, the microscopic control for each robot is computed to track the GMM trajectory (Line 10). The process of SwarmPRM at the macroscopic planning stage is shown in Fig. 2.

A. Macroscopic Motion Planning in GMM Space

Consider the macroscopic planning problem in GMM space $\mathcal{GM} \triangleq \{\varrho \mid \varrho = \sum_{i=1}^k \omega_i g_i, \forall g_i \in \mathcal{G}, k \in \mathbb{N}, \sum_{i=1}^k \omega_i = 1\}$. Denote the initial and target swarm state as $\varrho_{T_0} = \sum_{i=1}^{N_1} \omega^i_{T_0} g^i_{T_0}, \varrho_{T_f} = \sum_{j=1}^{N_2} \omega^j_{T_f} g^j_{T_f}$, respectively, where ϱ_{T_0} comprises N_1 Gaussian components $\Gamma_{T_0} = \{g^1_{T_0}, \cdots, g^{N_1}_{T_0}\}$ with weights $\boldsymbol{\omega}_{T_0} = [\omega^1_{T_0}, \cdots, \omega^{N_1}_{T_0}]$, and ϱ_{T_f} consists of N_2 Gaussian components $\Gamma_{T_f} = \{g^1_{T_f}, \cdots, g^{N_2}_{T_f}\}$ with weights $\boldsymbol{\omega}_{T_f} = [\omega^1_{T_f}, \cdots, \omega^{N_2}_{T_f}]$. Building upon the discussions in Sec. III, we assume the optimal transport between ϱ_{T_0} and ϱ_{T_f} can be achieved through a set of Gaussian trajectories $\Xi^* = \{\xi^*_{i,j}(t), t \in [T_0, T_f], i \in \{1, \cdots, N_1\}, j \in \{1, \cdots, N_2\}\}$, each $\xi^*_{i,j}(t)$ originating from the ith Gaussian component of ϱ_{T_0} and ultimately reaching the jth Gaussian component of ϱ_{T_f}. To satisfy the normalization property of the GMM, each Gaussian trajectory $\xi^*_{i,j}(t)$ is assigned a weight $\lambda^*_{i,j}$ satisfying $\sum_{i,j} \lambda^*_{i,j} = 1$, so that the optimal GMM trajectory can be derived as $\chi(t) = \sum_{i,j} \lambda^*_{i,j} \xi^*_{i,j}(t)$,

and the optimal transport cost can be computed as the sum of the transport cost along each Gaussian trajectory $\xi^*_{i,j}(t)$, which is determined by the Wasserstein metric and weighted by $\lambda^*_{i,j}$.

The SwarmPRM approach is presented in Alg. 2. The function RoadmapConstruction constructs a Gaussian roadmap $G = (V, E)$ including n nodes $g_1, \cdots, g_n$ and the set of Gaussian components of the initial and target GMMs (Line 1-2), as detailed in Sec. III. Through employing graph search on the Gaussian roadmap, the trajectory $\hat{\xi}_{i,j}(t)$ with the lowest transport cost $\mathcal{L}(g^i_{T_0}, g^j_{T_f})$ between each pair of Gaussian distributions $(g^i_{T_0}, g^j_{T_f})$ can be calculated (Line 3-8), which is an approximation to the computationally intractable optimal Gaussian trajectory $\xi^*_{i,j}(t)$. Therefore, we can provide an approximate solution to the optimization problem (1) in GMM space through computing the optimal GMM trajectory on the Gaussian roadmap. Detailedly, we compute the optimal weights $\lambda_{i,j}$ allocated to individual Gaussian trajectories $\hat{\xi}_{i,j}(t)$ to obtain the GMM trajectory $\sum_{i,j} \lambda_{i,j}\hat{\xi}_{i,j}(t)$ with minimal cost, which can be modeled as a linear programming (LP) problem (Line 9)

$$\min_{\Lambda} \quad \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} \lambda_{i,j}\mathcal{L}(g^i_{T_0}, g^j_{T_f}) \tag{13}$$

$$\text{s.t.} \quad \sum_{i=1}^{N_1} \lambda_{i,j} = \omega^j_{T_f}, \forall j \in \{1, \cdots, N_2\}, \tag{13a}$$

$$\sum_{j=1}^{N_2} \lambda_{i,j} = \omega^i_{T_0}, \forall i \in \{1, \cdots, N_1\}, \tag{13b}$$

where $\Lambda \triangleq \{\lambda_{i,j}, i \in \{1, \cdots, N_1\}, j \in \{1, \cdots, N_2\}\}$ denotes the transport policy.

Remark 1. *The Gaussian trajectories planned by SwarmPRM may overlap during certain time intervals,* i.e.*, there may exist* $T_1, T_2, \hat{\xi}_{i_1,j_1}, \hat{\xi}_{i_2,j_2}, T_0 \leqslant T_1 < T_2 \leqslant T_f, \forall t \in [T_1, T_2], \hat{\xi}_{i_1,j_1}(t) = \hat{\xi}_{i_2,j_2}(t)$. *The number of Gaussian components of the GMM trajectory* $\chi(t)$ *is thus time-varying.*

B. Microscopic Motion Planning

Upon determining the optimal evolution of the macroscopic state, represented by the time-varying GMM, each robot can utilize it to compute the collision-free trajectory from the initial position to the target area (Line 10). A computationally efficient artificial potential field (APF) approach is adopted to compute the microscopic control inputs for all N robots,

$$\boldsymbol{u}_i = -\frac{\partial(w_1 U_{att} + w_2 U_{rep})}{\partial \boldsymbol{u}_i}, i = 1, \cdots, N, \tag{14}$$

where U_{att}, U_{rep} represent the attractive and repulsive potentials, respectively, and w_1, w_2 are pre-defined weights representing the desired tradeoff between attractive and repulsive objectives. The detailed implementation of the APF approach can be found in [5].

Algorithm 2: SwarmPRM Algorithm

Input: ϱ_{T_0} : the initial GMM, ϱ_{T_f} : the target GMM, $\mathcal{Q}$: the initial positions of swarm robots, $\boldsymbol{\mathcal{O}}$: the environmental obstacles, n : the number of nodes in the Gaussian roadmap, r : the connection radius
Output: $\mathcal{T}$: the trajectories of swarm robots
1: Obtain the set of Gaussian components $\Gamma_{T_0}, \Gamma_{T_f}$ of initial and target GMMs ϱ_{T_0} and ϱ_{T_f}
2: $G =$ RoadmapConstruction$(n, r, \boldsymbol{\mathcal{O}}, \Gamma_{T_0} \cup \Gamma_{T_f})$
3: **for all** $g^i_{T_0} \in \Gamma_{T_0}$ **do**
4: **for all** $g^j_{T_f} \in \Gamma_{T_f}$ **do**
5: Compute the shortest path $\hat{\xi}_{i,j}(t)$ from $g^i_{T_0}$ to $g^j_{T_f}$ on graph G using graph search
6: Compute the transport cost $\mathcal{L}(g^i_{T_0}, g^j_{T_f})$
7: **end for**
8: **end for**
9: Obtain Λ by solving the LP in (13)
10: Compute $\mathcal{T}$ from $\mathcal{Q}$, $\{\hat{\xi}_{i,j}(t)\}$, and Λ by solving (14)

V. Simulation Results

This section evaluates the effectiveness of the proposed SwarmPRM approach via several simulations. First, two challenging cluttered environments are used to benchmark SwarmPRM against three representative approaches both qualitatively and quantitatively. Subsequently, the effectiveness of the risk measure CVaR on collision avoidance is assessed.

A. Simulation Setup

In each simulation scenario, the swarm is tasked with navigating from a given initial distribution $\varrho_{T_0} = \sum_{i=1}^{N_1} \omega^i_{T_0} g^i_{T_0}$ to a target distribution $\varrho_{T_f} = \sum_{j=1}^{N_2} \omega^j_{T_f} g^j_{T_f}$, while avoiding collisions with obstacles in $\mathcal{W} = [0, W] \times [0, H]$, where $W = 200$m, $H = 160$m, $N_1 = 4$, $N_2 = 3$, $\omega^1_{T_0} = \frac{1}{4}$, $\omega^2_{T_0} = \frac{3}{8}$, $\omega^3_{T_0} = \frac{3}{16}$, $\omega^4_{T_0} = \frac{3}{16}$, $\omega^1_{T_f} = \frac{1}{4}$, $\omega^2_{T_f} = \frac{3}{8}$, $\omega^3_{T_f} = \frac{3}{8}$. All Gaussian components of ϱ_{T_0} and ϱ_{T_f} share the same covariance matrix, $100I_2$, where I_2 denotes the second-order identity matrix. The means of $g^i_{T_0}, i \in \{1, 2, 3, 4\}$ and $g^j_{T_f}, j \in \{1, 2, 3\}$ are $[25, 20]$, $[25, 40]$, $[25, 120]$, $[25, 140]$, and $[175, 40]$, $[175, 60]$, $[175, 120]$, respectively. Each robot is of circular shape with a radius of 0.2m and is characterized by single-integrator dynamics

$$\dot{\boldsymbol{x}}_i(t) = \boldsymbol{u}_i(t), \boldsymbol{x}_i(T_0) = \boldsymbol{q}_i, \forall i \in \{1, \cdots, N\}, \tag{15}$$

where $\boldsymbol{x}_i = [x_i, y_i]^T$ denotes the robot position, and the control input $\boldsymbol{u}_i$ is a vector of linear velocities in the x- and y-directions. When constructing the Gaussian roadmap G, we set the number of samples $n = 500$, and the connection radius $r = 20$. The simulations are implemented using MATLAB code, and executed on a laptop with Intel Core i7-1065G7 CPU@1.30GHz and 16GB RAM. In practical implementation, we integrate the uniform sampler with a Gaussian sampler [21] to achieve a better coverage of the difficult area in the collision-free space. Furthermore, to avoid excessively high local densities in the swarm macroscopic state which adds difficulty to microscopic planning,

TABLE I: Simulation Results in Environment I

N	SwarmPRM	Formation control	dRRT*	ADOC
20	4.9 / **250.3**	**4.8** / 296.0	120.0 (36.2) / 259.6 (273.6)	8.0 / 271.0
40	**5.2** / **253.4**	8.0 / 309.2	120.0 (102.2) / 466.0 (466.0)	8.2 / 271.1
100	**5.8** / **254.0**	18.7 / 320.3	– / –	8.8 / 272.8
500	**11.6** / **255.7**	81.5 / 327.9	– / –	15.8 / 274.1

TABLE II: Simulation Results in Environment II

N	SwarmPRM	Formation control	dRRT*	ADOC
20	4.7 / 192.3	**2.8** / 271.2	120.0 (35.5) / **180.3** (270.4)	10.6 / 205.8
40	**5.0** / **192.7**	5.6 / 272.8	120.0 (109.4) / 399.4 (399.4)	10.9 / 206.0
100	**5.5** / **193.1**	14.7 / 280.4	– / –	11.5 / 206.8
500	**11.0** / **193.5**	68.8 / 280.7	– / –	19.6 / 208.5

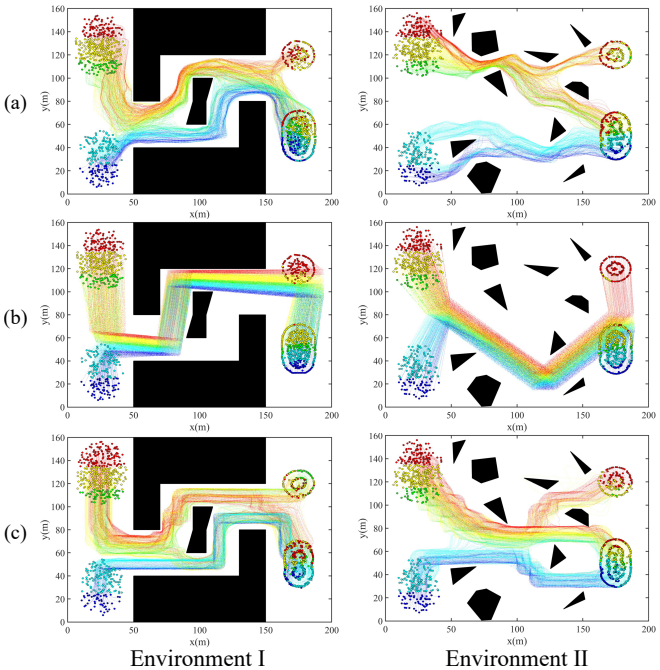

Fig. 3: **Trajectory comparison in environment I and II.** The trajectories of the swarm robotic system comprised of $N = 500$ robots obtained by (a) SwarmPRM, (b) Formation control, and (c) ADOC, for the same set of initial positions in each environment. The initial and final positions are represented by colored circles on the left and right sides of each subfigure, respectively, while the obstacles are depicted in black color.

we impose an upper limit on the probability density of the GMM $\chi(t)$ solved by the LP problem (13), which can be formulated as a minimum-cost flow problem [22].

B. Results and Analyses

Two representative two-dimensional complex environments are designed to evaluate the performance of our approach both qualitatively and quantitatively, as well as three state-of-the-art benchmark methods, including multi-robot formation control [23], discrete RRT* [9], and adaptive distributed optimal control [5], denoted as Formation control, dRRT*, and ADOC, respectively. In each environment, swarm robotic systems consisting of $N = 20, 40, 100, 500$ robots are required to move from initial configurations to target areas. The performance can be assessed by the computational time T_{sol} (min) and the average trajectory length $\overline{D}$ (m),

$$\overline{D} = \frac{1}{N}\sum_{i=1}^{N}\sum_{\tau=0}^{\kappa-1}\|\boldsymbol{x}_i[T_0+(\tau+1)\Delta t]-\boldsymbol{x}_i(T_0+\tau\Delta t)\|_2, \quad (16)$$

where $\kappa = \frac{T_f-T_0}{\Delta t}$. The quantitative results are shown in Tab. I and Tab. II, with the values before and after the slash representing T_{sol} and $\overline{D}$, respectively. We set the computational time limit of dRRT* to be 2 hours, and record T_{sol} and $\overline{D}$ when dRRT* stops. Furthermore, we document the computational time required to find the first solution and its coresponding $\overline{D}$ in the bracket.

1) Environment I: We consider an environment containing 3 nonconvex obstacles, which is presented in the left column of Fig. 3. All approaches successfully plan a collision-free trajectory for each robot with respect to all obstacles and other robots. As shown in Tab. I, our SwarmPRM approach surpasses other methods in terms of T_{sol} and $\overline{D}$ in almost all scenarios. In comparison with ADOC, which plans the time-varying GMM using a predetermined set of collocation Gaussian components, our approach demonstrates superior performance in both metrics, showcasing the efficacy of the sampling-based method on generating GMM trajectories with lower costs within shorter computational time. Regarding Formation control, the global planner necessitates adhering to the formation for the entirety of the swarm, leading to an increased average trajectory length compared to other methods. Moreover, the local motion planning entails solving distributed nonlinear optimizations for each robot at high frequency to avoid collisions, resulting in a rapid increase in the overall computational time T_{sol} as the swarm size grows, which exceeds that of SwarmPRM in scenarios where $N = 40, 100, 500$. In comparison with dRRT*, our approach computes solutions with shorter T_{sol} and lower $\overline{D}$ in all cases thanks to the hierarchical planning strategy. The dRRT* faces a substantial computational burden and can only find solutions for swarm robotic systems consisting of up to around 40 robots in environment I within the allotted two-hour computational time, which is insufficient for finding a solution with $\overline{D}$ better than our approach. Furthermore, we notice that the computational time required by SwarmPRM to plan the GMM trajectory at the macroscopic stage is independent of the swarm size N, showcasing good scalability of our approach to large-scale swarm robots.

The left column of Fig. 3 shows the trajectories of the swarm robotic system comprising 500 robots planned by SwarmPRM, Formation control, and ADOC, respectively. The SwarmPRM and ADOC allow splitting and merging of the swarm's PDF and plan trajectories with lower $\overline{D}$ in comparison with the Formation control which requires the maintenance of a formation for all robots in the swarm.

2) Environment II: In environment II, 10 obstacles are randomly distributed. The results, as shown in Tab. II and the right column of Fig. 3, show similar trends to those discussed in Sec. V-B.1. The qualitative result shows that the macro-

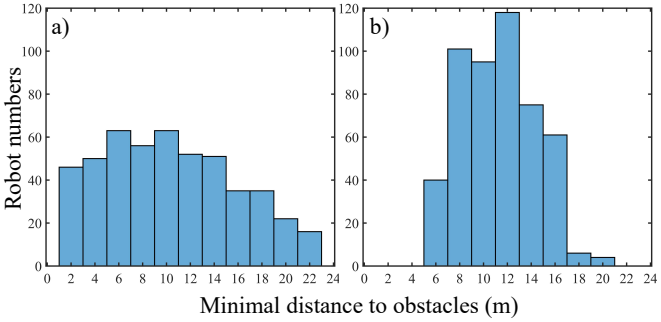

Fig. 4: (a) and (b) illustrate the minimal distance from each robot to the set of obstacles along the overall trajectory, with risk tolerance level $\alpha = 0.3$ and 0.1, respectively.

scopic trajectory planned by SwarmPRM can cross "narrow passages", leading to shorter average trajectory length $\overline{D}$ compared to ADOC, Formation control, and dRRT*.

C. Evaluation of the Risk Measure CVaR

To demonstrate the effectiveness and risk awareness of the risk measure CVaR, we set the swarm size to be 500, and the risk tolerance level α to be 0.3 and 0.1. During the construction of the Gaussian roadmap, the functions INFREE and COLLISIONFREE ensure the conditional expectation of the SDF within the 30% and 10% worst-case quantiles to fall in the safe region, respectively. We calculate the minimal distance from each robot to the set of obstacles along the entire trajectory under different values of the risk tolerance level α, and visualize the number of robots falling into different distance intervals. As Fig. 4 shows, a smaller risk tolerance level α leads to a higher percentage of robots positioned further away from the obstacles, affirming the effectiveness and risk awareness of employing CVaR for collision avoidance. Furthermore, we can regulate the proximity of the robot swarm to obstacles by adjusting the risk tolerance level α.

VI. CONCLUSION

We develop the SwarmPRM motion planning approach, offering a new perspective on developing sampling-based motion planning methods for large-scale swarm robotic systems. We propose to construct a risk-aware Gaussian roadmap, where each node represents a Gaussian distribution, and the distance and geodesic path between nodes are defined based on the Wasserstein metric. Furthermore, we incorporate CVaR for collision checking, enabling the generation of the risk-aware Gaussian roadmap. We formulate a linear programming problem to calculate the optimal GMM trajectory on the roadmap. Simulation results show that SwarmPRM significantly outperforms other state-of-the-art benchmark approaches in computational efficiency and average trajectory length. Moreover, the risk awareness and scalability of our approach are also validated. Future work includes extension to three-dimensional environments and implementation of SwarmPRM on real robotic platforms.

REFERENCES

[1] J. Keller, D. Thakur, M. Likhachev, J. Gallier, and V. Kumar, "Coordinated path planning for fixed-wing uas conducting persistent surveillance missions," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 17–24, 2016.

[2] K. McGuire, C. De Wagter, K. Tuyls, H. Kappen, and G. C. de Croon, "Minimal navigation solution for a swarm of tiny flying robots to explore an unknown environment," *Science Robotics*, vol. 4, no. 35, p. eaaw9710, 2019.

[3] D. S. Drew, "Multi-agent systems for search and rescue applications," *Current Robotics Reports*, vol. 2, pp. 189–200, 2021.

[4] 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.

[5] 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.

[6] 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.

[7] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[8] M. Čáp, P. Novák, J. Vokrínek, and M. Pěchouček, "Multi-agent rrt: sampling-based cooperative pathfinding," in *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, 2013, pp. 1263–1264.

[9] R. Shome, K. Solovey, A. Dobson, D. Halperin, and K. E. Bekris, "drrt*: Scalable and informed asymptotically-optimal multi-robot motion planning," *Autonomous Robots*, vol. 44, no. 3-4, pp. 443–467, 2020.

[10] K. Rudd, G. Foderaro, P. Zhu, and S. Ferrari, "A generalized reduced gradient method for the optimal control of very-large-scale robotic systems," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1226–1232, 2017.

[11] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath, "Coherent measures of risk," *Mathematical Finance*, vol. 9, no. 3, pp. 203–228, 1999.

[12] A. Hakobyan and I. Yang, "Distributionally robust risk map for learning-based motion planning and control: A semidefinite programming approach," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 718–737, 2022.

[13] 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*. Springer, 2023, pp. 513–525.

[14] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[15] L. Janson, B. Ichter, and M. Pavone, "Deterministic sampling-based motion planning: Optimality, complexity, and performance," *The International Journal of Robotics Research*, vol. 37, no. 1, pp. 46–61, 2018.

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

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[18] 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.

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

[20] H. Gao, P. Wu, Y. Su, K. Zhou, J. Ma, H. Liu, and C. Liu, "Probabilistic visibility-aware trajectory planning for target tracking in cluttered environments," *arXiv preprint arXiv:2306.06363*, 2023.

[21] V. Boor, M. H. Overmars, and A. F. Van Der Stappen, "The gaussian sampling strategy for probabilistic roadmap planners," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 2. IEEE, 1999, pp. 1018–1023.

[22] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, "Network flows," 1988.

[23] 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.