Simulation-Informed Diffusion for Decentralized Multi-robot Motion Planning

Jinhao Liang
University of Virginia
njs4nu@virginia.edu

Sven Koenig
University of California, Irvine
sven.koenig@uci.edu

Ferdinando Fioretto
University of Virginia
fioretto@virginia.edu

Abstract

Decentralized multi-robot motion planning requires each robot to generate collision-free trajectories from local observations, without global sensing or reliable communication. However, most existing planners, whether classical or learning-based, generate trajectories from a static snapshot of the local observation, which limits their ability to anticipate the future behavior of neighboring robots. This limitation is critical as the number of robots increases and the environment becomes more cluttered. Motivated by world models that use future state estimation to improve planning decisions, this paper introduces *Simulation-Informed Diffusion* (SID), a decentralized framework built on constraint-aware diffusion models (CADM) that simulate neighboring robots' future trajectories to guide each robot's own planning. SID first uses CADM to simulate the future trajectories of neighboring robots from their currently observed states, and then uses the same CADM to plan each robot's own trajectory under safety constraints informed by these simulations. Crucially, the accurate simulation of neighbors enables a minimal communication scheme that triggers coordination only when necessary in highly congested scenarios. Experiments across diverse environments show that SID consistently outperforms baseline methods in terms of planning effectiveness and constraint satisfaction, and scales to scenarios with 108 robots and 160 obstacles.

1 Introduction

Multi-Robot Motion Planning (MRMP) seeks collision-free trajectories for multiple robots operating in a shared environment, and underpins applications ranging from automated warehouses to drone fleets and autonomous driving systems [2]. Many MRMP methods assume a centralized planner with full access to all robot states [17, 29]. However, this assumption is often violated in practical systems, where communication may be limited or unreliable, and each robot may operate under partial observability of the environment [7]. The key challenge in such settings is to generate collision-free trajectories from local observations with minimal communication.

A common way to address this decentralized planning problem is to compute feasible actions or trajectories based on the current local observation. In particular, *reactive methods*, such as ORCA [34], select an immediate feasible action by reasoning over the current positions and velocities of nearby robots and obstacles. These methods are efficient and decentralized, but they are inherently myopic: a feasible action computed from a single snapshot ignores future interactions and may fail in dense environments. To mitigate this issue, *learning-based methods* have been proposed to learn mappings from local observations to actions or future states [28, 36]. These methods can capture more complex relationships between observations and actions, extending the reactive paradigm but inherit its myopia, since plans still derive from the current observation alone. The importance of anticipating future states for planning is also supported by recent work on world models [6, 15, 38]. Finally, *prediction-based planning* methods predict future trajectories of neighboring robots and use these as planning references [39, 19]. These methods provide forward-looking signals, but the limited accuracy of the

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

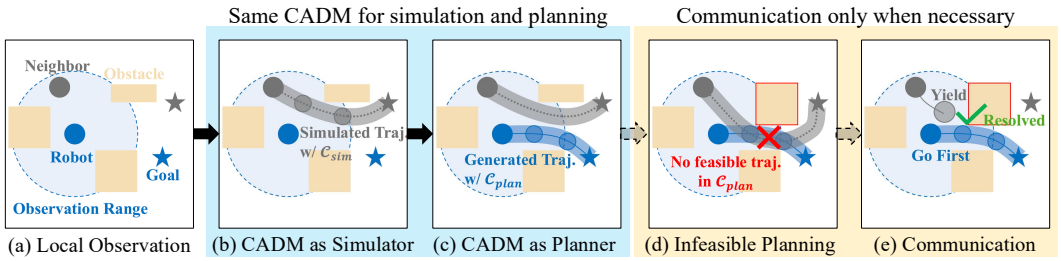


Figure 1: Overview of SID. From a local observation, the robot uses CADM first to simulate feasible future trajectories for neighboring robots and then to plan its own trajectory under safety constraints. If the local planner cannot find a feasible trajectory under the current simulated constraints, SID triggers communication among the affected robots to coordinate actions.

predictions may cause substantial mismatch with the trajectories neighbors actually execute. This mismatch ultimately undermines the reliability of these references for decentralized planning.

To overcome these challenges, this paper introduces *Simulation-Informed Diffusion (SID)*, a decentralized framework in which each robot uses a single *constraint-aware diffusion model* (CADM) as both simulator and planner. The proposed constraint-aware diffusion model generates trajectories that satisfy collision-avoidance and kinematic constraints; it augments standard diffusion sampling with a proximal term and uses a Lagrangian dual method to solve the resulting constrained optimization problem. As a simulator, CADM generates accurate future trajectories for neighboring robots from their current observed states. As a planner, it combines these simulated trajectories with the current local observation to generate a collision-free trajectory by projecting diffusion samples onto the planning constraint set. Crucially, the accuracy of this constraint-aware diffusion-based simulator enables each robot to greatly minimize communication. The proposed communication scheme triggers messages only when local planning cannot resolve a conflict, eliminating unnecessary exchange in sparse conditions. In particular, SID develops a communication mechanism driven by simulation-informed conflict detection. Robots exchange messages only when local simulation and planning cannot resolve predicted conflicts, reducing communication while maintaining planning performance. In an extensive MRMP benchmark of increasing size and complexity, SID outperforms representative reactive, learning-based, and prediction-based planning methods (ORCA [34], IA-MPC [39], and MIMIC-D [10]) in trajectory quality and constraint satisfaction, and in large-scale congested scenarios it remains effective using up to 108 robots and 160 obstacles. Figure 1 summarizes the SID framework.

2 Related Work

Multi-robot Motion Planning. Multi-robot motion planning has traditionally been formulated as a joint planning problem, where a centralized planner computes collision-free trajectories for all robots. Classical approaches typically instantiate this view through optimization-based formulations [3, 26, 13]. These methods can reason about global feasibility and robot-robot interactions, but they require a central planner that collects all robot states, solves a coupled planning problem, and distributes commands. This creates a single point of failure [30] and requires reliable communication, which is difficult to maintain in distributed environments [37]. These limitations motivated decentralized MRMP, where each robot plans from local observations and limited communication. In such a setting, *reactive collision-avoidance methods* [34, 11, 1] removed the need for a central planner by allowing each robot to select locally safe actions from nearby robot states and obstacles; however, given their myopic nature, they often fail in dense environments where future interactions determine safety. *Learning-based decentralized planners* extend this local decision-making paradigm by learning policies or planning rules from local observations [28, 9, 36]. These methods improve expressiveness over hand-designed reactive rules, but many remain tied to snapshot-based inputs [35]. More recent methods predict future trajectories of neighboring robots and use them as planning references [39, 19]. This line of work introduces an important capability: local planners can reason about where nearby robots may be over the planning horizon, rather than only where they are now. However, standard neural predictors are not themselves constrained planners. As a result, they can produce trajectories that are dynamically infeasible, nonsmooth, or inconsistent with the trajectories that neighboring

robots will actually execute. SID addresses this limitation by replacing unconstrained prediction with constraint-aware simulation, generating feasible future references that are aligned with the planner's trajectory distribution.

Generative Models for Motion Planning. Generative models have recently provided a different route for motion planning: instead of searching for a single trajectory under a hand-designed cost, they learn distributions over feasible or high-quality trajectories. This is particularly attractive in continuous motion planning, where the solution space is multimodal and where a robot may need to choose among qualitatively different homotopy classes. Such approaches have been applied to single-robot motion planning [5, 21, 25, 24] and centralized MRMP [20, 29], with projection and discrete guidance improving safety and scalability [16, 17]. Recent work has also explored decentralized MRMP. However, they either condition each robot only on local observations [40, 10], which limits their ability to anticipate future neighbor behavior, or require fixed-frequency communication among robots [14], which increases communication burden. SID closes this gap by using the same constraint-aware diffusion model as a local simulator and planner. Each robot simulates feasible future trajectories for its neighbors, conditions its own planning on these simulated futures, and communicates only when simulation-informed planning cannot resolve a predicted conflict.

3 Preliminaries

The paper focuses on decentralized multi-robot motion planning under partial observability and limited communication. We first introduce the problem formulation, then review generative diffusion models, which are the foundation of our proposed method.

Decentralized multi-robot motion planning. Consider a set $\mathcal{A} = \{1, \dots, N\}$ of N robotic agents operating within an environment $\mathcal{W} \subset \mathbb{R}^2$. The environment contains a set of static obstacles $\mathcal{O}$, defining the *free space* as $\mathcal{F} := \mathcal{W} \backslash \mathcal{O}$. Each robot $i \in \mathcal{A}$ starts from an initial state $\mathbf{s}_i$ and aims to reach a goal state $\mathbf{g}_i$. Unlike centralized MRMP, where a planner observes the full joint state of all robots, in the decentralized setting each robot plans from local information alone. Robot i receives a local observation $\mathbf{o}_i$ encoding the portion of the environment and neighboring robots visible within its sensing range [7]. When communication is available, robot i may also receive messages $\mathbf{c}_i = \{\mathbf{m}_{j\to i}\}_{j\in\mathcal{N}_i}$ from neighboring robots in its communication neighborhood $\mathcal{N}_i$ [36]. The decentralized planner π_θ aims to map these local inputs to a collision-free trajectory of horizon H. This planned trajectory for robot i is denoted as $\boldsymbol{\tau}_i = (\mathbf{p}_i^0, \mathbf{p}_i^1, \dots, \mathbf{p}_i^H)$. The planner is defined as:

$$\boldsymbol{\tau}_i = \pi_\theta(\mathbf{o}_i, \mathbf{c}_i, \mathbf{g}_i). \tag{1}$$

Following a standard convention in MRMP, this work models each robot i as a disk of radius r_i and assumes a maximum speed $v_{\max}$ [29, 16]. A set of trajectories $\{\boldsymbol{\tau}_i\}_{i\in\mathcal{A}}$ is feasible if it satisfies obstacle avoidance, inter-robot separation, and kinematic constraints at every planning step:

$$\begin{aligned}
&\operatorname{dist}(\mathbf{p}_i^\psi, \mathcal{O}) \geq r_i, \quad \forall i \in \mathcal{A},\ \psi \in [H], \\
&\|\mathbf{p}_i^\psi - \mathbf{p}_j^\psi\|_2 \geq r_i + r_j; \quad \forall i, j \in \mathcal{A},\ i \neq j,\ \psi \in [H], \\
&\|\mathbf{p}_i^{\psi+1} - \mathbf{p}_i^\psi\|_2 \leq v_{\max}, \quad \forall i \in \mathcal{A},\ \psi \in [H{-}1],
\end{aligned} \tag{2}$$

where ψ denotes the planning step. The first constraint requires each robot to remain in the obstacle-free region with clearance at least its radius. The second enforces pairwise separation between robots. The third imposes a maximum displacement between consecutive planning steps, which acts as a speed limit. SID uses these constraints both to guide trajectory generation and to determine when local simulation is insufficient and communication is required.

Generative diffusion models for trajectory planning. Generative diffusion models [23, 12] produce high-fidelity complex data by operating in two phases: a *forward step*, that gradually introduces noise to clean data samples, followed by a *reverse denoising step*, in which a deep neural network is trained to iteratively remove noise. This forward process defines a Markov chain $\{\boldsymbol{x}_t\}_{t=0}^T$, with initial sample $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$ and each transition $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ realized by adding Gaussian noise according to a variance schedule α_t. In trajectory planning, the clean sample is a trajectory over a finite horizon, e.g., $\boldsymbol{x}_0 \equiv \boldsymbol{\tau}_i = (\mathbf{p}_i^0, \mathbf{p}_i^1, \dots, \mathbf{p}_i^H)$, or a vectorized representation of it. In *score-based diffusion* [32, 33], the reverse process relies on a learned score function $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) = \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)$ which points toward

regions of higher probability under the noisy distribution. This score function is trained to minimize

$$\min_{\theta} \mathbb{E}_{\substack{t\sim[1,T],p(\boldsymbol{x}_0),\\ q(\boldsymbol{x}_t|\boldsymbol{x}_0)}}(1-\alpha_t)\left[\left\|\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)-\nabla_{\boldsymbol{x}_t}\log q(\boldsymbol{x}_t|\boldsymbol{x}_0)\right\|^2\right]. \tag{3}$$

where α_t is a variance schedule that controls the noise level at each step. Once trained, the model generates a trajectory by initializing $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and iteratively denoising it. This is also known as the *sampling* phase, which commonly follows the Stochastic Gradient Langevin Dynamics update rule [32, 22]:

$$\boldsymbol{x}_t = \boldsymbol{x}_{t+1} + \frac{\epsilon}{2}\boldsymbol{s}_\theta(\boldsymbol{x}_{t+1}, t+1) + \sqrt{\epsilon}\boldsymbol{z}, \tag{4}$$

where ϵ denotes the step size and $\boldsymbol{z}$ is Gaussian noise.

4 Simulation-Informed Diffusion for Decentralized MRMP

This section presents Simulation-Informed Diffusion (SID), a diffusion-based framework for decentralized MRMP. We first describe the Constraint-aware Diffusion Model (CADM), which generates feasible trajectories under safety constraints. We then use CADM as a local simulator to predict feasible future trajectories for neighboring robots. Finally, we present a simulation-guided diffusion planner that conditions each robot's trajectory generation on these simulated futures, together with a minimal communication mechanism that activates only when local planning becomes infeasible.

Constraint-aware Diffusion Models. Diffusion models can capture complex trajectory distributions, but they may still produce infeasible trajectories, limiting their use in robotic systems [5, 16, 18]. To enforce collision-avoidance and kinematic constraints during sampling, we build on projected diffusion methods for motion planning [8, 16] and define the constrained target density at diffusion step t as $\tilde{p}_t(\boldsymbol{x}_t) \propto p_t(\boldsymbol{x}_t)\mathbb{I}(\boldsymbol{x}_t \in \mathcal{C})$, where $\mathcal{C}$ denotes the feasible set induced by the planning constraints. Unlike standard diffusion sampling, which seeks only high-probability samples under p_t, this formulation seeks high-probability trajectories while requiring every diffusion iterate to lie in $\mathcal{C}$. Operationally, CADM modifies the SGLD update in Eq. 4 to include a projection step:

$$\boldsymbol{x}_t = \mathcal{P}_{\mathcal{C}}\left(\boldsymbol{x}_{t+1} + \frac{\epsilon}{2}\boldsymbol{s}_\theta(\boldsymbol{x}_{t+1}, t+1) + \sqrt{\epsilon}\boldsymbol{z}\right), \tag{5}$$

where $\mathcal{P}_{\mathcal{C}}(\boldsymbol{x}_t) = \arg\min_{\boldsymbol{y}\in\mathcal{C}} \|\boldsymbol{y} - \boldsymbol{x}_t\|_2^2$ is the Euclidean projection operator onto the constraint set $\mathcal{C}$. This projection enforces feasibility at every sampling step and guides the diffusion model toward constraint-satisfying trajectories.

Given the non-convexity of $\mathcal{C}$, this projection is non-trivial. However, the structure of the constraints allows for efficient optimization. The obstacle avoidance and inter-robot separation constraints can be expressed as differentiable functions of the trajectory, while the kinematic constraints are simple norm bounds on consecutive trajectory points. We write $\mathcal{C} = \{\boldsymbol{y} \mid \mathbf{h}(\boldsymbol{y}) \leq \mathbf{0}\}$, where $\mathbf{h}(\boldsymbol{y})$ stacks the collision-avoidance and kinematic inequality constraints, and solve the resulting projection problem using the augmented Lagrangian:

$$\mathcal{L}_\rho(\boldsymbol{y}, \boldsymbol{\nu}) = \|\boldsymbol{y} - \boldsymbol{x}_t\|_2^2 + \boldsymbol{\nu}^\top \mathbf{h}(\boldsymbol{y}) + \frac{\rho}{2}\|\max(\mathbf{0}, \mathbf{h}(\boldsymbol{y}))\|_2^2, \tag{6}$$

where $\boldsymbol{\nu} \geq \mathbf{0}$ are the Lagrange multipliers and $\rho > 0$ is the penalty parameter. The projection subproblem is solved by leveraging the classical primal-dual optimization method: we alternate gradient descent on the primal variable $\boldsymbol{y}$ and dual ascent on $\boldsymbol{\nu}$, with update $\boldsymbol{\nu} \leftarrow \max(\mathbf{0}, \boldsymbol{\nu} + \rho\,\mathbf{h}(\boldsymbol{y}))$ [27, 4]. After each round, ρ is increased. Projection stops when the maximum constraint violation falls below 10^{-5} or after 100 primal–dual iterations. If projection fails at an intermediate denoising step, we retain the minimum-violation iterate and continue sampling. If the final trajectory remains infeasible, we identify the bottleneck set and trigger communication-based replanning.

Neighbor Trajectory Simulation. Even if trajectory feasibility can be enforced by the CADM sampler, in decentralized MRMP, the inter-robot constraints in Eq. 2 depend on the future positions of neighboring robots over the planning horizon. If these future positions are unavailable, the planning robot can only constrain its trajectory against the currently observed neighbor states, which reduces CADM to a reactive collision-avoidance planner. SID avoids this limitation by using CADM not only to plan the agent's trajectory, but also to simulate feasible future trajectories for observed neighbors. These simulated trajectories define the time-indexed references against which each robot

enforces inter-robot safety. Formally, robot i applies the simulator π_θ independently to each observed neighbor:

$$\hat{\boldsymbol{\tau}}_j = \pi_\theta(\mathbf{o}_i, \mathbf{c}_i, \mathbf{g}_j), \quad \forall j \in \mathcal{N}_i, \tag{7}$$

where $\hat{\boldsymbol{\tau}}_j = (\hat{\mathbf{p}}_j^0, \hat{\mathbf{p}}_j^1, \ldots, \hat{\mathbf{p}}_j^H)$ denotes the simulated trajectory of neighbor j over the planning horizon H. We assume neighboring goals $\mathbf{g}_j$ are provided at initialization. Thus, the neighboring goal $\mathbf{g}_j$ is known and does not require repeated online communication. The key role of this step is not merely to predict where neighbor j may move, but to produce a realistic trajectory reference generated by the same constraint-aware diffusion mechanism used for planning, as shown in Figure 1(b).

To serve as a reliable planning reference, $\hat{\boldsymbol{\tau}}_j$ must satisfy obstacle avoidance, kinematic limits, and inter-robot separation. When simulating neighbor j, robot i treats the stored planned trajectories of the other observed neighbors from the previous planning round as fixed references; denote these trajectories by $\boldsymbol{\tau}_l = (\mathbf{p}_l^0, \ldots, \mathbf{p}_l^H)$. The resulting simulation constraint set is:

$$\mathcal{C}_{\text{sim}}(j) = \left\{ \hat{\boldsymbol{\tau}}_j \;\middle|\; \begin{array}{ll} \operatorname{dist}(\hat{\mathbf{p}}_j^h, \mathcal{O}) \geq r_j, & h \in [H], \\ \|\hat{\mathbf{p}}_j^{h+1} - \hat{\mathbf{p}}_j^h\|_2 \leq v_{\max}, & h \in [H-1], \\ \|\hat{\mathbf{p}}_j^h - \mathbf{p}_l^h\|_2 \geq r_j + r_l, & \forall l \in \mathcal{N}_i \setminus \{j\},\ h \in [H] \end{array} \right\}. \tag{8}$$

Simulation-guided Diffusion-based Decentralized Planning. Given the simulated neighboring trajectories $\{\hat{\boldsymbol{\tau}}_j\}_{j \in \mathcal{N}_i}$, robot i uses them as dynamic references for planning its trajectory, which is illustrated in Figure 1(c). We adopt a receding-horizon control (RHC) scheme for decentralized planning [40, 10]. At planning time ψ, robot i acquires a local observation $\mathbf{o}_i^\psi$, simulates the future trajectories of observed neighbors, and then generates a trajectory $\boldsymbol{\tau}_i = (\mathbf{p}_i^0, \mathbf{p}_i^1, \ldots, \mathbf{p}_i^H)$. Although the planner optimizes over all H future steps, it executes only the first k, where $1 \leq k < H$. After execution, robot i receives a new local observation $\mathbf{o}_i^{\psi+k}$, re-simulates the neighboring trajectories $\{\hat{\boldsymbol{\tau}}_j^{\psi+k}\}_{j \in \mathcal{N}_i^{\psi+k}}$, and replans over the next horizon. This observation-simulation-planning-execution loop lets the local plan adapt as interactions evolve.

To guarantee safety in each planning round, the planned trajectory for robot i must satisfy environmental obstacle avoidance, kinematic limits, and inter-robot safety with respect to the simulated neighboring trajectories. The planning constraint set can be written as:

$$\mathcal{C}_{\text{plan}}(i) = \left\{ \boldsymbol{\tau}_i \;\middle|\; \begin{array}{ll} \operatorname{dist}(\mathbf{p}_i^h, \mathcal{O}) \geq r_i, & h \in [H], \\ \|\mathbf{p}_i^{h+1} - \mathbf{p}_i^h\|_2 \leq v_{\max}, & h \in [H-1], \\ \|\mathbf{p}_i^h - \hat{\mathbf{p}}_j^h\|_2 \geq r_i + r_j,\ \forall j \in \mathcal{N}_i, & h \in [H] \end{array} \right\}. \tag{9}$$

The set $\mathcal{C}_{\text{plan}}(i)$ enforces feasibility with respect to environmental obstacles and simulated neighboring trajectories, but it does not determine when robot i should reach its goal. In diffusion-based planning, the travel time is typically predefined because the planner samples trajectories over a fixed horizon [29, 16]. However, a fixed arrival schedule is restrictive in decentralized MRMP: robots in sparse regions should reach their goals sooner, whereas robots in congested regions may need more time to avoid neighbors safely.

Algorithm 1 Simulation-guided Decentralized Planning

Input: Robot i, goal $\mathbf{g}_i$, CADM π_θ, RHC window k
Initialize planning time $\psi \leftarrow 0$
while robot i not at $\mathbf{g}_i$ **do**
  Acquire $\mathbf{o}_i$ and observed neighbors $\mathcal{N}_i$
  for each neighbor $j \in \mathcal{N}_i$ **do**
    Simulate $\hat{\boldsymbol{\tau}}_j$ with π_θ under $\mathcal{C}_{\text{sim}}(j)$
  end for
  Define $\mathcal{C}_{\text{plan}}(i)$ and $\widetilde{\mathcal{C}}_{\text{plan}}(i)$ using $\{\hat{\boldsymbol{\tau}}_j\}_{j \in \mathcal{N}_i}$
  Define $\mathcal{C}_{\text{hold}}(i)$ using space-time A* on $\widetilde{\mathcal{C}}_{\text{plan}}(i)$
  Generate $\boldsymbol{\tau}_i$ with π_θ under $\mathcal{C}_{\text{plan}}(i) \cap \mathcal{C}_{\text{hold}}(i)$
  Execute the first k time steps of $\boldsymbol{\tau}_i$
  Advance planning time $\psi \leftarrow \psi + k$
end while

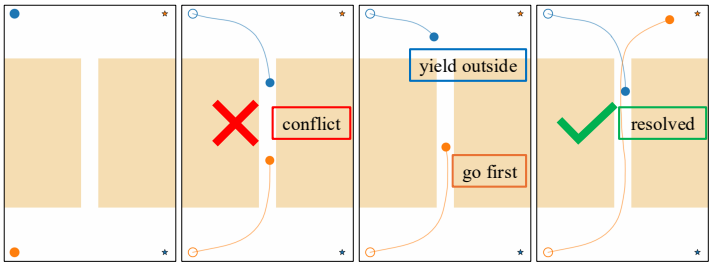
To address this issue, we estimate an adaptive arrival time for each robot at each RHC round and impose it as a goal-holding constraint, allowing the planner to reduce travel time while preserving safe goal reaching. Specifically, at each planning round, robot i runs a space-time A* search on a discretized version of the local planning problem under $\mathcal{C}_{\text{plan}}(i)$ [31]. Let $\widetilde{\mathcal{C}}_{\text{plan}}(i)$ denote the resulting discretized feasible set. The search returns the earliest feasible arrival step $a_i = \min\{h \in \{0, \ldots, H\} \mid \tilde{\mathbf{p}}_i^h = \mathbf{g}_i\}$.

Because space-time A* operates on a discretized feasible set, a_i may be later than the shortest feasible arrival time in the continuous trajectory space used by SID. We therefore treat a_i as a conservative arrival step. Given a_i, we define goal-holding constraints

$$\mathcal{C}_{\text{hold}}(i) = \left\{ \boldsymbol{\tau}_i \,\middle|\, \mathbf{p}_i^h = \mathbf{g}_i, \quad h = a_i, \ldots, H \right\}. \tag{10}$$

Overall, robot i plans under the combined feasible set $\mathcal{C}_{\text{plan}}(i) \cap \mathcal{C}_{\text{hold}}(i)$. This combined constraint keeps the trajectory safe with respect to obstacles and simulated neighbors while ensuring that it reaches the goal by the conservative target step and remains there afterward. The complete procedure is summarized in Algorithm 1.

Communication Mechanism Design. Simulation-guided planning resolves most local interactions by using simulated future trajectories as references for local trajectory generation. However, some crowded interactions require explicit coordination rather than simulation alone, as shown in Figure 1(d)–(e). Consider a case in which several robots may try to enter a narrow passage that admits only one robot at a time. Local observations and simulated trajectories alone cannot determine a passing order, as illustrated in Figure 2, panels (a) and (b). In such cases, robots need communication to share planned trajectories and break the symmetry of the interaction.



(a) Initial state (b) Coordination Conflict (c) High-priority Plan (d) Low-priority Plan

Figure 2: Example of a narrow corridor bottleneck.

To address this issue while minimizing communication, we add a minimal communication mechanism on top of the simulation-guided planner. Accurate simulation and simulation-guided planning give SID a concrete trigger for intervention: if Algorithm 1 cannot find a feasible trajectory under $\mathcal{C}_{\text{plan}}(i)$, then robot i has reached a coordination bottleneck that local simulation alone cannot resolve. The mechanism therefore remains inactive during routine planning and activates only when local planning becomes infeasible. When this trigger fires, robot i first identifies the robots involved in the unresolved local interaction. We denote this bottleneck set by $\mathcal{B}_i$. Specifically, the set $\mathcal{B}_i$ contains robot i and observed neighbors whose inter-robot collision-avoidance residual exceeds 10^{-5} at any step of the planning horizon. SID then assigns priority within $\mathcal{B}_i$ using each robot's remaining distance to goal, $d_j = \|\mathbf{p}_j - \mathbf{g}_j\|_2$ for $j \in \mathcal{B}_i$. Robots with smaller d_j receive higher priority, which yields a deterministic passing order, as illustrated in Figure 2, panels (c) and (d).

Algorithm 2 Minimal Communication Mechanism

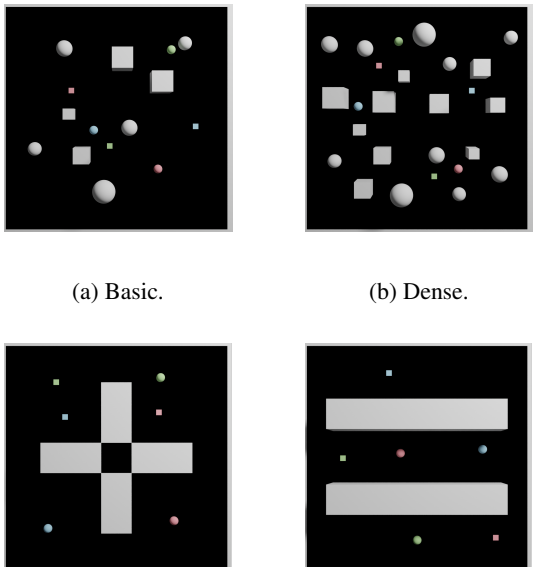
Input: Robot i, planning constraint $\mathcal{C}_{\text{plan}}(i)$, neighbors $\mathcal{N}_i$
Identify bottleneck set $\mathcal{B}_i \subseteq \mathcal{N}_i \cup \{i\}$
Compute $d_j = \|\mathbf{p}_j - \mathbf{g}_j\|_2$ for all $j \in \mathcal{B}_i$
Rank robots in $\mathcal{B}_i$ by ascending d_j
Remove inter-robot constraints involving $\mathcal{B}_i \setminus \{i\}$ from $\mathcal{C}_{\text{plan}}(i)$
if i is not first in this order **then**
    Receive one trajectory $\boldsymbol{\tau}_j$ from each earlier robot j
    Add $\boldsymbol{\tau}_j$ to $\mathcal{C}_{\text{plan}}(i)$ for each earlier robot j
end if
Generate $\boldsymbol{\tau}_i$ under the updated $\mathcal{C}_{\text{plan}}(i)$
Broadcast $\boldsymbol{\tau}_i$ once to later robots in $\mathcal{B}_i$
return $\boldsymbol{\tau}_i$

The fallback planner replans robots in $\mathcal{B}_i$ according to this priority order, as shown in Algorithm 2. Before replanning, each robot removes from its planning constraint set the inter-robot constraints associated with robots in $\mathcal{B}_i \setminus \{i\}$. The first robot in the order therefore plans without constraints from later robots and broadcasts its trajectory once. Each later robot waits for the planned trajectories of all earlier robots. For each earlier robot j, the current robot adds the communicated trajectory $\boldsymbol{\tau}_j$ to its planning constraint set and replans under the updated constraints. After replanning, it broadcasts its own trajectory to later robots in $\mathcal{B}_i$. Thus, communication is restricted to the bottleneck set, and

each robot transmits at most one trajectory during a fallback episode. The communicated trajectories convert an ambiguous simultaneous interaction into an ordered planning problem, resolving the symmetry that caused planning infeasibility.

5 Experiments

Settings. This paper evaluates decentralized MRMP algorithms in two complementary settings. **(1)** The first setting tests **Feasibility, Scalability, and Quality** on a modified MRMP benchmark adapted from [16]. Each map has spatial extent 2×2. The benchmark includes four representative map types: random maps with varying obstacle densities (basic and dense), and structured maps that mimic real-world environments (room and shelf layouts), as shown in Figure 3. Colorful spheres and plates denote the starts and goals of robots, and white objects indicate obstacles. We evaluate each map type with 3, 6, and 9 robots across 25 different configurations. To test scalability beyond the original benchmark, we further extend the basic and dense maps to larger teams with 12, 15, and 18 robots. The local observation range is measured as a fraction of the map size and is set to 0.5 by default. The receding-horizon controller executes $k = 8$ steps at each planning round, and the diffusion model outputs trajectories with a maximum length of 64 steps. We further analyze sensitivity to the observation range and execution window. **(2)** The second setting evaluates SID on **challenging scenarios**. These scenarios test scalability under severe congestion, including large-scale instances with 108 robots and 160 obstacles.



(a) Basic. (b) Dense.

(c) Room. (d) Shelf.

Figure 3: Examples of benchmark maps used for MRMP experiments.

Evaluation Metrics. We evaluate each algorithm using Success rate, Communication Frequency, Smoothness, and Travel time. A test case is successful only if every robot reaches its goal and no collision occurs at any time. *Success rate* is the percentage of successful test cases. *Communication Frequency* is the total number of communication events in an instance divided by the number of robots, averaged over test instances. *Smoothness* is computed from the squared discrete acceleration along each trajectory and reported as the mean per-agent accumulated value. *Travel time* is the mean arrival time across all robots in a successful instance. *Smoothness* and *Travel time* are computed only on successful test cases.

Competing Methods. We compare SID with three competitive decentralized baselines that cover the main families of multi-robot coordination methods:

1. **Optimal Reciprocal Collision Avoidance (ORCA)** is a classical decentralized reactive planner that selects collision-avoiding velocities under reciprocal responsibility assumptions [34].
2. **Interaction-Aware Model Predictive Control (IA-MPC)** combines learned interaction-aware trajectory prediction with model predictive control, providing a predictive optimization baseline for dynamic multi-robot interactions [39].
3. **Multi-agent Coordination with Decentralized Diffusion Policies (MIMIC-D)** learns decentralized diffusion policies for multi-agent coordination through multimodal imitation, providing a state-of-the-art generative planning baseline [10].

All methods use the same observation range and execution window during evaluation. All neural-network-based methods are trained on the same dataset. The details of the implementation, additional results, and visualizations for challenging scenarios are provided in Appendices A and B. We also provide videos of the generated trajectories in the supplementary material.

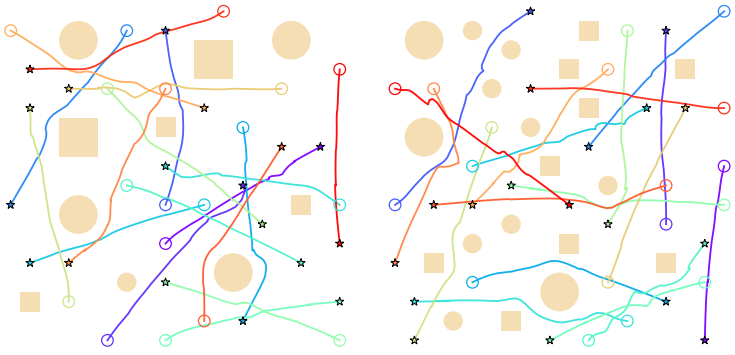
Map	Robots	ORCA				IA-MPC				MIMIC-D				SID			
		S	C	A	T	S	C	A	T	S	C	A	T	S	C	A	T
Basic	3	88	0	0.24	21.92	36	8	0.15	63.00	8	0	0.47	63.00	**100**	0.05	1.06	41.11
	6	64	0	0.28	22.89	4	8	0.08	63.00	0	0	N/A	N/A	**100**	0.04	1.23	41.94
	9	52	0	0.32	23.33	0	8	N/A	N/A	0	0	N/A	N/A	**100**	0.07	1.03	42.35
Dense	3	96	0	0.26	23.35	20	8	0.16	63.00	12	0	0.65	63.00	**100**	0	1.17	41.45
	6	64	0	0.30	24.27	0	8	N/A	N/A	0	0	N/A	N/A	**100**	0.06	1.16	42.49
	9	28	0	0.33	25.25	0	8	N/A	N/A	0	0	N/A	N/A	**100**	0.06	1.12	42.48
Room	3	68	0	0.20	22.02	52	8	0.13	63.00	40	0	0.59	63.00	**100**	0.08	0.81	39.93
	6	36	0	0.24	23.28	4	8	0.06	63.00	24	0	0.38	63.00	**100**	0.88	1.09	40.05
	9	20	0	0.27	22.62	0	8	N/A	N/A	12	0	0.26	63.00	**84**	3.08	0.97	39.92
Shelf	3	64	0	0.16	20.94	52	8	0.12	29.82	52	0	0.56	63.00	**100**	0.08	0.68	42.83
	6	28	0	0.22	21.71	4	8	0.07	41.67	36	0	0.37	63.00	**92**	0.92	1.15	41.89
	9	12	0	0.26	21.78	0	8	N/A	N/A	8	0	0.24	63.00	**96**	1.12	1.02	41.29

Table 1: Results for the original benchmark robot counts. S denotes success rate (%), C denotes communication frequency, A denotes acceleration-based smoothness, and T denotes travel time measured by average arrival time. Smoothness and travel time are calculated within feasible instances. Values in column A are reported in units of 10^{-2}.

5.1 Comparison across Methods

Table 1 evaluates each method at the robot counts used in the original benchmarks. *ORCA* degrades significantly as robot count and map complexity increase. Specifically, its success drops from 88% on Basic maps with three robots to 28% on Dense maps, 20% on Room maps, and 12% on Shelf maps with nine robots. This drop reflects a limitation of reactive planning, which uses only the current local observation and does not anticipate neighboring robots. *IA-MPC* performs worse. Although it uses an RNN to predict neighboring robots' future trajectories as references for decentralized planning, large prediction errors can mislead the planner, yielding zero success with nine robots across all four map types. *MIMIC-D* uses diffusion models as decentralized planners, but generated trajectories are not guaranteed to satisfy planning constraints in complex environments. This limitation is consistent with observations in diffusion-based centralized planning [29, 16]. Because *MIMIC-D* also plans from the current local observation, it cannot anticipate neighboring robots. It fails on Basic and Dense maps with nine robots and reaches only 12% and 8% success on Room and Shelf maps.

In contrast, SID achieves perfect success on Basic and Dense maps with up to nine robots, and reaches 84% and 96% success on Room and Shelf maps with nine robots. These results improve over the strongest baseline by **4.2**× and **8.0**× on the two structured maps. SID also requires much less communication than IA-MPC. IA-MPC communicates before every planning round to collect neighboring trajectory histories, whereas SID simulates neighbors from the current local observation and communicates only when coordination is needed. Thus, SID's communication frequency stays close to that of communication-free methods. SID has higher acceleration because acceleration is measured only on feasible trajectories, and harder successful cases often require more complex motions; the absolute values remain small. Finally, unlike the fixed-horizon arrival schedule used by IA-MPC and MIMIC-D, SID uses $\mathcal{C}^{\psi}_{\text{hold}}(i)$ to reduce per-robot travel time.



(a) Basic map. (b) Dense map.

Figure 4: Trajectories generated by SID with 18 robots.

Map	Robots	ORCA				IA-MPC				MIMIC-D				SID			
		S	C	A	T	S	C	A	T	S	C	A	T	S	C	A	T
Basic	12	28	0	0.35	24.57	0	8	N/A	N/A	0	0	N/A	N/A	**100**	0.08	1.12	42.48
	15	4	0	0.37	25.93	0	8	N/A	N/A	0	0	N/A	N/A	**92**	0.20	0.92	42.21
	18	0	0	N/A	N/A	0	8	N/A	N/A	0	0	N/A	N/A	**80**	0.33	0.96	41.90
Dense	12	8	0	0.35	25.83	0	8	N/A	N/A	0	0	N/A	N/A	**92**	0.21	1.08	42.51
	15	8	0	0.36	26.43	0	8	N/A	N/A	0	0	N/A	N/A	**96**	0.15	0.98	42.44
	18	4	0	0.38	27.94	0	8	N/A	N/A	0	0	N/A	N/A	**88**	0.41	1.05	41.57

Table 2: Scalability results on the extended basic and dense maps.

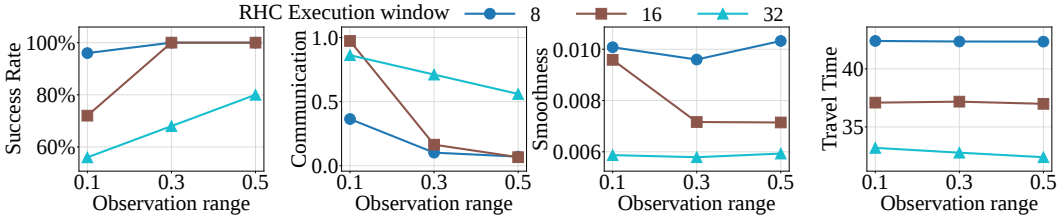


Figure 5: Sensitivity of SID on Basic maps with 9 robots. We sweep the local observation range and the execution window k in receding-horizon control.

Table 2 shows that SID scales beyond the baselines as team size increases. IA-MPC and MIMIC-D solve no extended cases. ORCA succeeds in only a few settings. As the team grows from 12 to 18 robots, ORCA drops from 28% to 0% on Basic maps and from 8% to 4% on Dense maps. SID maintains high success across 12, 15, and 18 robots: 100%, 92%, and 80% on Basic maps, and 92%, 96%, and 88% on Dense maps. At 18 robots, SID is the only successful method on Basic maps and exceeds ORCA by 84 percentage points on Dense maps. SID communicates sparsely, with frequency at most 0.41 compared with the fixed value of eight for IA-MPC. Figure 4 shows representative 18-robot trajectories generated by SID.

5.2 Sensitivity Analysis

Figure 5 studies the sensitivity of SID to two deployment parameters: the local observation range and the receding-horizon execution window k. The observation range primarily affects the quality of local interaction modeling. With a larger neighborhood, each robot can simulate more relevant neighbors, which improves success and reduces communication. At $k = 16$, success increases from 72% at observation range 0.1 to 100% at ranges 0.3 and 0.5, while communication decreases from 0.97 to 0.16 and 0.07. The execution window k controls the feedback frequency. Smaller windows replan more often and are therefore more robust to interaction uncertainty, while larger windows execute longer trajectory segments before replanning. In the tested settings, increasing k from 8 to 32 reduces travel time from approximately 42.4 steps to 32.4–33.2 steps and reduces acceleration from approximately 0.010 to 0.006, but also lowers success from 96–100% to 56–80%. These trends show that the gains from longer execution windows are not free: they reduce feedback and can make the planner less robust. For the Basic-map benchmark with nine robots, $k = 16$ with observation range 0.3 or 0.5 achieves the most balanced empirical behavior among the tested configurations, combining 100% success, low communication, and shorter travel time than the more conservative $k = 8$ setting.

5.3 Ablation Studies

To examine the roles of neighbor trajectory simulation and communication, we conduct ablation studies on the original benchmark and settings. Table 3 reports success rates for 3, 6, and 9 robots over 25 instances per setting.

With nine robots, SID achieves 100% success on Basic and Dense maps and 84% and 96% on Room and Shelf maps, respectively. In contrast, reactive CADM (*NoS+NoC*) achieves only 20%, 8%, 8%, and 4% across the same maps. Even with communication, the constant-velocity variant (*NoS+C*)

Map	Basic			Dense			Room			Shelf		
Robots	3	6	9	3	6	9	3	6	9	3	6	9
NoS+NoC	84	56	20	56	20	8	72	36	8	88	40	4
NoS+C	**100**	76	28	80	44	16	88	60	20	88	56	4
S+NoC	92	92	72	**100**	92	76	92	76	48	96	88	60
S+C (SID)	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**84**	**100**	**92**	**96**

Table 3: Ablation success rates (%) on the original benchmark. *S+C* is full SID; *NoS+C* uses constant-velocity references with communication; *S+NoC* retains CADM simulation but removes fallback communication; and *NoS+NoC* uses CADM purely reactively from constant-velocity references.

achieves only 28%, 16%, 20%, and 4%, respectively. This means constant-velocity references may miss future conflicts, showing that communication alone cannot compensate for inaccurate neighbor references. Moreover, removing communication (*S+NoC*) reduces success rates from SID's 100%, 100%, 84%, and 96% to 72%, 76%, 48%, and 60% on Basic, Dense, Room, and Shelf maps with nine robots, respectively, showing that simulation alone cannot resolve bottleneck deadlocks. These results indicate that simulation-informed planning contributes substantially to SID's performance gains beyond those provided by the CADM backbone or communication alone, while communication remains important for resolving coordination bottlenecks.

5.4 Challenging Scenarios

We further evaluate SID in large-scale environments that combine multiple obstacle patterns within a single map. These scenarios contain 108 robots and 160 obstacles, creating dense interactions and narrow passages. The robots have start–goal pairs with varying travel distances, requiring robots with short and long travel paths to coordinate under partial observations. To the best of our knowledge, no existing diffusion-based decentralized MRMP method has been demonstrated on environments of this scale and complexity. Figure 6 in Appendix B shows all 108 trajectories alongside a view highlighting eight robots with longer travel paths.

6 Conclusion

This paper introduced Simulation-Informed Diffusion (SID), a decentralized MRMP framework that uses a constraint-aware diffusion model both to simulate neighboring robots and to plan each robot's own trajectory under safety constraints. By replacing snapshot-based reactive planning with simulated future references, SID accounts for spatial and temporal interactions over the planning horizon. The same simulation signal also supports a minimal communication mechanism that activates only at coordination bottlenecks where local planning becomes infeasible. Across diverse settings and challenging 108-robot scenarios, SID improves success rates over reactive, predictive, and diffusion-policy baselines while keeping communication sparse.

Limitations and future work. While SID provides state-of-the-art results in 2D decentralized MRMP tasks, several directions remain open. A natural next step is to extend the framework to 3D manipulation. Moreover, extending the proposed framework to settings where constraints are specified implicitly remains a promising direction for future work. Neighbor simulation also introduces additional computational overhead. Future work can consider faster constrained sampling.

Acknowledgment

This research at the University of Virginia was partially supported by NSF grants 2334936, 2334448, NSF CAREER Award 2401285, and DARPA Contract No. #HR0011252E005. The research at the University of California, Irvine was supported by NSF grants 2544613, 2434916, 2321786, 2112533, as well as gifts from Amazon Robotics and the Donald Bren Foundation. The authors acknowledge Research Computing at the University of Virginia for providing computational resources that have contributed to the results reported within this paper. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies, or the U.S. government. The authors declare no competing interests.

References

[1] Javier Alonso-Mora, Andreas Breitenmoser, Martin Rufli, Paul Beardsley, and Roland Siegwart. Optimal reciprocal collision avoidance for multiple non-holonomic robots. In *Distributed autonomous robotic systems: The 10th international symposium*, pages 203–216. Springer, 2013.

[2] Luke Antonyshyn, Jefferson Silveira, Sidney Givigi, and Joshua Marshall. Multiple mobile robot task and motion planning: A survey. *ACM Computing Surveys*, 55(10):1–35, 2023.

[3] Federico Augugliaro, Angela P. Schoellig, and Raffaello D'Andrea. Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1917–1922, 2012.

[4] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[5] João Carvalho, An Thai Le, Piotr Kicki, Dorothea Koert, and Jan Peters. Motion planning diffusion: Learning and adapting robot motion planning with diffusion models. *IEEE Transactions on Robotics*, 41:4881–4901, 2025. doi: 10.1109/TRO.2025.3593109.

[6] Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

[7] Yuxiao Chen, Ugo Rosolia, and Aaron D. Ames. Decentralized task and path planning for multi-robot systems. *IEEE Robotics and Automation Letters*, 6(3):4337–4344, 2021. doi: 10.1109/LRA.2021.3068103.

[8] Jacob K Christopher, Stephen Baek, and Ferdinando Fioretto. Constrained synthesis with projected diffusion models. In *Advances in Neural Information Processing Systems*, volume 37. Curran Associates, Inc., 2024.

[9] Mehul Damani, Zhiyao Luo, Emerson Wenzel, and Guillaume Sartoretti. Primal$_2$: Pathfinding via reinforcement and imitation multi-agent learning - lifelong. *IEEE Robotics and Automation Letters*, 6(2):2666–2673, 2021. doi: 10.1109/LRA.2021.3062803.

[10] Dayi Dong, Maulik Bhatt, Seoyeon Choi, and Negar Mehr. Mimic-d: Multi-modal imitation for multi-agent coordination with decentralized diffusion policies. *arXiv preprint arXiv:2509.14159*, 2025.

[11] Ke Guo, Dawei Wang, Tingxiang Fan, and Jia Pan. Vr-orca: Variable responsibility optimal reciprocal collision avoidance. *IEEE Robotics and Automation Letters*, 6(3):4520–4527, 2021. doi: 10.1109/LRA.2021.3067851.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[13] Akshay Jaitly, Jack Cline, and Siavash Farzan. A milp-based solution to multi-agent motion planning and collision avoidance in constrained environments. In *2025 IEEE 21st International Conference on Automation Science and Engineering (CASE)*, pages 2200–2207. IEEE, 2025.

[14] Jeric Lew, Yuhong Cao, Derek Ming Siang Tan, and Guillaume Sartoretti. Aid: Agent intent from diffusion for multi-agent informative path planning. *arXiv preprint arXiv:2512.02535*, 2025.

[15] Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

[16] Jinhao Liang, Jacob K Christopher, Sven Koenig, and Ferdinando Fioretto. Simultaneous multi-robot motion planning with projected diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 37162–37180. PMLR, 13–19 Jul 2025.

[17] Jinhao Liang, Sven Koenig, and Ferdinando Fioretto. Discrete-guided diffusion for scalable and safe multi-robot motion planning. *arXiv preprint arXiv:2508.20095*, 2025.

[18] Xiaoxuan Liang, Saeid Naderiparizi, Yunpeng Liu, Berend Zwartsenberg, and Frank Wood. Improved constrained generation by bridging pretrained generative models. *arXiv preprint arXiv:2603.06742*, 2026.

[19] Lars Lindemann, Matthew Cleaveland, Gihyun Shim, and George J. Pappas. Safe planning in dynamic environments using conformal prediction. *IEEE Robotics and Automation Letters*, 8 (8):5116–5123, 2023. doi: 10.1109/LRA.2023.3292071.

[20] Hao Luan, Yi Xian Goh, See-Kiong Ng, and Chun Kai Ling. Projected coupled diffusion for test-time constrained joint generation. *arXiv preprint arXiv:2508.10531*, 2025.

[21] Yunhao Luo, Chen Sun, Joshua B Tenenbaum, and Yilun Du. Potential based diffusion motion planning. *arXiv preprint arXiv:2407.06169*, 2024.

[22] Chenrui Ma. Transition flow matching. *arXiv preprint arXiv:2603.15689*, 2026.

[23] Chenrui Ma, Xi Xiao, Tianyang Wang, Xiao Wang, and Yanning Shen. Learning straight flows: Variational flow matching for efficient generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 38154–38164, 2026.

[24] Chenrui Ma, Xi Xiao, Lin Zhao, Xinru Jiang, Tianyang Wang, Yanning Shen, and Ferdinando Fioretto. Posterior flow matching. *arXiv preprint arXiv:2610.13994*, 2026.

[25] Chenrui Ma, Xi Xiao, Lin Zhao, Tianyang Wang, Ferdinando Fioretto, and Yanning Shen. Drift flow matching. *arXiv preprint arXiv:2605.17244*, 2026.

[26] Tobia Marcucci, Jack Umenberger, Pablo Parrilo, and Russ Tedrake. Shortest paths in graphs of convex sets. *SIAM Journal on Optimization*, 34(1):507–532, 2024.

[27] Parikh Neal, Chu Eric, Peleato Borja, and Eckstein Jonathan. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[28] Guillaume Sartoretti, Justin Kerr, Yunfei Shi, Glenn Wagner, T. K. Satish Kumar, Sven Koenig, and Howie Choset. Primal: Pathfinding via reinforcement and imitation multi-agent learning. *IEEE Robotics and Automation Letters*, 4(3):2378–2385, 2019. doi: 10.1109/LRA.2019.2903261.

[29] Yorai Shaoul, Itamar Mishani, Shivam Vats, Jiaoyang Li, and Maxim Likhachev. Multi-robot motion planning with diffusion models. *arXiv preprint arXiv:2410.03072*, 2024.

[30] Weihua Sheng, Qingyan Yang, Jindong Tan, and Ning Xi. Distributed multi-robot coordination in area exploration. *Robotics and autonomous systems*, 54(12):945–955, 2006.

[31] David Silver. Cooperative pathfinding. In *Proceedings of the aaai conference on artificial intelligence and interactive digital entertainment*, volume 1, pages 117–122, 2005.

[32] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[33] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[34] Jur Van Den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics research: the 14th international symposium ISRR*, pages 3–19. Springer, 2011.

[35] Rose Wang, J. Chase Kew, Dennis Lee, Tsang-Wei Lee, Tingnan Zhang, Brian Ichter, Jie Tan, and Aleksandra Faust. Model-based reinforcement learning for decentralized multiagent rendezvous. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 711–725. PMLR, 16–18 Nov 2021. URL `https://proceedings.mlr.press/v155/wang21d.html`.

[36] Yutong Wang, Bairan Xiang, Shinan Huang, and Guillaume Sartoretti. Scrimp: Scalable communication for reinforcement- and imitation-learning-based multi-agent pathfinding. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9301–9308, 2023. doi: 10.1109/IROS55552.2023.10342305.

[37] Adam Wiktor, Dexter Scobee, Sean Messenger, and Christopher Clark. Decentralized and complete multi-robot motion planning in confined spaces. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1168–1175, 2014. doi: 10.1109/IROS.2014.6942705.

[38] Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[39] Hai Zhu, Francisco Martinez Claramunt, Bruno Brito, and Javier Alonso-Mora. Learning interaction-aware trajectory predictions for decentralized multi-robot motion planning in dynamic environments. *IEEE Robotics and Automation Letters*, 6(2):2256–2263, 2021. doi: 10.1109/LRA.2021.3061073.

[40] Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. Madiff: Offline multi-agent learning with diffusion models. *Advances in Neural Information Processing Systems*, 37:4177–4206, 2024.

A Implementation Details

A.1 Software and Hardware

Software used for experiments is Ubuntu 22.04.5, Python 3.8.20, CUDA 12.1, and PyTorch 2.1.2. For each of our experiments, we used 1 NVIDIA RTX A6000 GPU.

A.2 Training Details

Our implementation builds upon the official code of Liang et al. [16] and Shaoul et al. [29], with modifications to accommodate our specific requirements. Table 4 shows the hyperparameters used for training diffusion models in our experiments.

Table 4: Hyperparameters for Training in Experiments.

HyperParameters	Value
Diffusion Sampling Step	25
Learning Rate	1e-4
Batch Size	64
Optimizer	Adam

A.3 Planning Details.

All experiments are conducted on 2×2 maps. The robot radius was set to 0.03 units across four different maps. Other parameters related to planning are summarized in Table 5.

Table 5: Simulation and planning parameters used in our experiments.

Parameter	Value
Map size	2×2 units
Robot radius	0.03 units
Maximum speed	$v_{\max} = 0.05$
Maximum Planning horizon	$H = 64$
Receding-horizon execution window	$k = 8$
Receding-horizon execution window in sensitivity analysis	$k = 8, 16, 32$
Maximum observation range	$0.5 \times$ map size
Maximum observation range in sensitivity analysis	$0.5, 0.3, 0.1 \times$ map size

A.4 Benchmark Details.

The benchmark consists of four map families: basic, dense, room, and shelf. Each type of map contains 25 test instances, where each instance specifies both the obstacle layout and the start–goal pairs. The map configurations are summarized in Table 6. For the standard benchmark, we evaluate 3, 6, and 9 robots on all four maps, resulting in $4 \times 3 \times 25 = 300$ test instances per method. For the extended scalability benchmark, we additionally evaluate 12, 15, and 18 robots on the basic and dense maps, resulting in another $2 \times 3 \times 25 = 150$ test instances per method.

Table 6: Map families used in the benchmark. Each map family contains 25 test instances.

Map family	Obstacle configuration	Obstacle size
Basic	10 randomly placed obstacles	0.1–0.2 units
Dense	20 randomly placed obstacles	0.1–0.2 units
Room	2 long rectangular obstacles	0.2×1.0 units
Shelf	2 long rectangular obstacles	0.2×1.0 units

A.5 Evaluation Metrics.

Success rate. An instance is counted as successful only if all robots reach their goals and the executed trajectories are collision-free with respect to both environmental obstacles and other robots. The reported success rate is

$$S = 100 \cdot \frac{\text{number of successful instances}}{\text{total number of instances}} \tag{11}$$

Communication frequency. For SID, communication is activated only when the simulation-guided planner cannot find feasible solutions. The instance-level communication frequency is

$$C_q = \frac{\text{total number of communication events}}{\text{number of robots}} \tag{12}$$

and the reported communication frequency is

$$C = \frac{\sum_1^Q C_q}{\text{total number of instances}}. \tag{13}$$

In the implementation, ORCA and MIMIC-D use no explicit communication, so their communication frequency is zero. IA-MPC needs to communicate at every receding-horizon planning round; with $H = 64$ and $k = 8$, this yields a fixed communication frequency of $H/k = 8$.

Smoothness. Smoothness is measured as the integral of squared discrete acceleration. For robot i in instance q, we compute

$$A_{i,q} = \Delta t \sum_{t=0}^{H-2} \left\| \frac{\mathbf{p}_{i,q}^{t+2} - 2\mathbf{p}_{i,q}^{t+1} + \mathbf{p}_{i,q}^{t}}{\Delta t^2} \right\|_2^2. \tag{14}$$

The instance-level smoothness is the mean across robots,

$$A_q = \frac{1}{N} \sum_{i=1}^{N} A_{i,q}. \tag{15}$$

We report the mean smoothness over successful instances. In the result tables, smoothness values are reported in units of 10^{-2}.

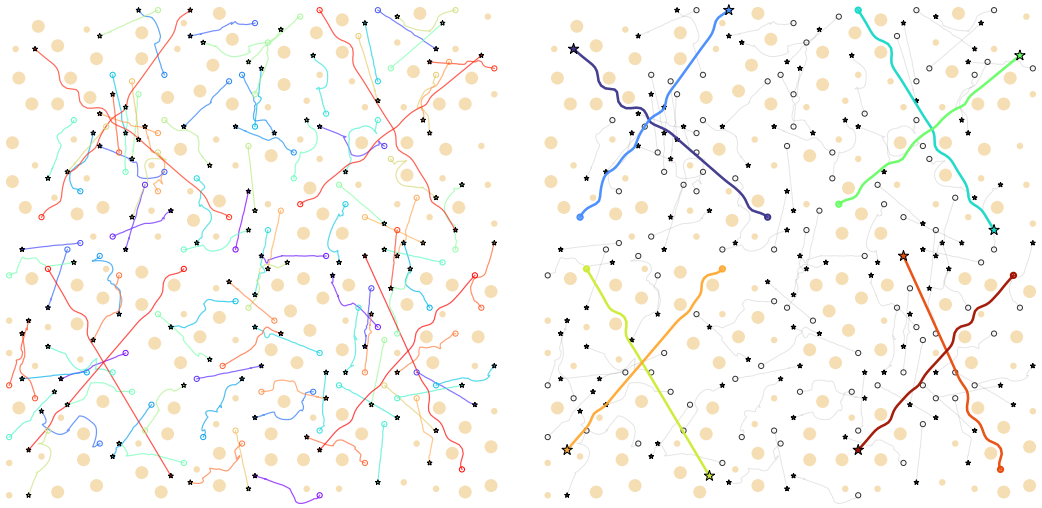
Travel time. For instance q, we define travel time as the average arrival time across robots. We report the mean travel time over successful instances.

B Additional Experimental Results

This section provides additional results that complement the main experimental evaluation. Table 7 reports the average running time of each method on successful instances. This should be interpreted together with the success rates in Table 1, since running time is averaged only over instances that are successfully solved. Specifically, methods with lower success rates may exhibit lower reported running times, as they tend to succeed primarily on easier instances that can be solved quickly. ORCA

Map	ORCA			IA-MPC			MIMIC-D			SID		
	3	6	9	3	6	9	3	6	9	3	6	9
Basic	0.11	0.35	0.68	48.24	38.45	N/A	7.52	N/A	N/A	76.01	125.04	175.92
Dense	0.34	0.87	1.75	115.50	N/A	N/A	4.89	N/A	N/A	140.03	205.73	261.74
Room	0.09	0.26	0.39	54.71	29.27	N/A	19.99	36.80	52.33	67.58	294.04	381.67
Shelf	0.10	0.29	0.51	58.96	33.01	N/A	19.37	28.24	51.41	94.24	406.32	552.94

Table 7: Running time on the original benchmark robot counts. Values are reported in seconds and averaged over successful instances only. N/A indicates that the method has no successful instance under the corresponding setting.



(a) All 108 generated trajectories.

(b) Trajectories of 8 robots with longer travel paths.

Figure 6: Representative trajectories generated by SID in a large-scale scenario with 108 robots and 160 obstacles. Left: all generated trajectories. Right: eight robots with longer travel paths are highlighted.

is the fastest method because it relies on lightweight reactive velocity selection, but its success rate decreases substantially in complex environments. IA-MPC and MIMIC-D often fail to solve any instances in harder settings. In particular, the running time of IA-MPC decreases as the number of robots increases on basic maps because the few instances it solves are comparatively simple. SID achieves the highest success rates while maintaining reasonable running times.

Figure 6 further visualizes SID in a large-scale scenario with 108 robots. The generated trajectories show that SID can coordinate a great number of robots through cluttered regions and narrow passages, illustrating the scalability of the proposed simulation-guided planning and communication mechanism beyond the standard benchmark sizes. The start–goal pairs span different travel distances, so robots with short and long travel paths must coordinate within the same environment. Figure 6a shows all 108 trajectories, while Figure 6b highlights eight robots with longer travel paths.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The results in experiments support the claims made in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include formal theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper describes the benchmark environments, robot counts, baseline methods, evaluation metrics, and key implementation details needed to reproduce the main experimental results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We plan to release the code and reproduction instructions upon publication.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the benchmark settings, evaluated robot counts, baselines, evaluation metrics, and key algorithmic parameters used in the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The current version reports aggregate performance over benchmark instances.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports the hardware setup and the compute resources used to run the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and believe the work conforms to it.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses potential impacts in the introduction.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Existing code and assets used in the experiments are properly credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The supplementary material includes anonymized videos illustrating the generated multi-robot trajectories and experimental scenarios.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.