ORION: Option-Regularized Deep Reinforcement Learning for Cooperative Multi-Agent Online Navigation

Shizhe Zhang*, Jingsong Liang*, Zhitao Zhou, Shuhan Ye, Yizhuo Wang, Derek Ming Siang Tan, Jimmy Chiun, Yuhong Cao, and Guillaume Sartoretti, *Member, IEEE*

***Abstract*—Existing methods for multi-agent navigation typically assume fully known environments, offering limited support for partially known scenarios with outdated or imperfect prior maps, such as warehouses or factory floors. There, agents need to balance path optimality with collecting and sharing environmental information to help teammates reach their own targets. To these ends, we propose *ORION*, a novel deep reinforcement learning framework for cooperative multi-agent online navigation in partially known environments. Starting from an imperfect prior map, ORION trains agents to make decentralized decisions, coordinate toward individual targets, and actively reduce task-relevant map uncertainty through online observation sharing in a closed perception—action loop. We first design a shared graph encoder that fuses prior map with online perception into a unified representation, providing robust state embeddings under environmental discrepancies. At the core of ORION is an option-critic framework that learns high-level cooperative modes translated into sequences of low-level actions, enabling adaptive switching between individual navigation and team-level exploration. We further introduce a dual-stage cooperation strategy that allows agents to assist teammates under map uncertainty, thereby reducing the overall makespan. Across extensive maze-like maps and large-scale warehouse environments, ORION achieves high-quality real-time decentralized cooperation while scaling to up to 10 robots, outperforming state-of-the-art classical and learning-based baselines. Finally, we validate ORION on physical robot teams, demonstrating its robustness and practicality for real-world cooperative navigation.**

***Index Terms*—AI-Based Methods, Path Planning for Multiple Mobile Robots or Agents, Distributed Robot Systems.**

I. Introduction

MULTI-AGENT navigation is a fundamental robotic problem, where a team of agents must reach individual targets while minimizing the overall makespan, i.e., the travel distance or time until the last agent reaches its assigned target. Beyond the challenges of single-agent navigation, such as long-horizon planning and dynamic obstacle avoidance, multi-agent navigation further introduces team-level cooperation: agents need to anticipate/react to dynamic interactions arising from neighboring robots and changes in shared environments.

Manuscript received December 30, 2025; Revised February 24, 2026; Accepted May 4, 2026. This paper was recommended for publication by Editor M. Ani Hsieh upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Temasek Laboratories (TL@NUS) under grant TL/FS/2025/01. *(Corresponding author: Yuhong Cao.)*

* Equal Contribution. The authors are with the Department of Mechanical Engineering, College of Design and Engineering, National University of Singapore, Singapore 117575 (email: {shizhezhang,jingsongliang,zhitao_zhou,e1373216,wy98,derektan,jimmy.chiun,caoyuhong}@u.nus.edu; mpegas@nus.edu.sg).

The code is available at https://github.com/marmotlab/ORION.

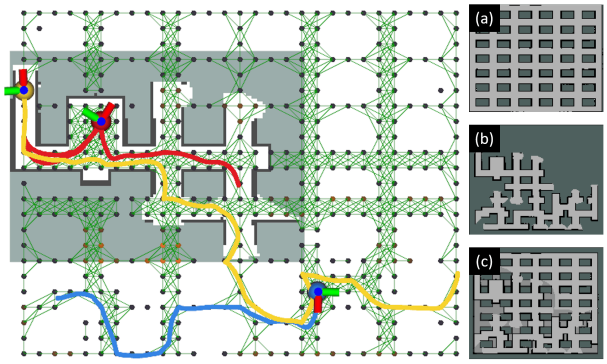

Fig. 1. Overview of Multi-Agent Online Navigation. Each agent is assigned a target and navigates over a prior map that may differ from the ground truth. Agents maintain/share (a) *prior map*, (b) *current map*, and (c) *combined map* that fuse prior/online sources to reason about partially changed environments. During navigation, agents not only pursue their own targets but also cooperate by sharing information and assisting others. For example, the red agent reaches its target early and then helps the yellow agent by exploring uncertain regions before returning. ORION enables such adaptive cooperation both before and after arrival on-goal, ultimately reducing the team's makespan by coordinating agents to contribute where they are most needed in a decentralized way.

However, many real-world deployments, especially in warehouses or factory floors, operate under imperfect or outdated prior maps due to frequent layout changes and occlusions. This motivates a nonstandard yet practical variant of multi-agent navigation, where robots must plan and coordinate online while jointly reducing map uncertainty through decentralized exploration and information sharing. This setting amplifies two key challenges: balancing individual target-reaching with team-level information gathering, and reasoning about dynamic multi-robot interactions under partial observability. In such cases, robots must navigate while continuously updating their belief of the map, requiring richer team-level reasoning that goes beyond classical (sometimes heuristic and offline) path planning. That is, environmental uncertainty forces the team to adaptively trade off between exploiting known areas and exploring unknown areas that may benefit the team's overall objective (e.g., find shortcuts for other agents, or help them avoid dead-ends). Cooperation is therefore more complex: a robot may need to temporarily deviate from its shortest path or intentionally gather information that primarily benefits others, sometimes at the expense of its own budget.

Early works in multi-agent navigation primarily focused on ensuring collision-free motion at the kinematic level, typically by decoupling target-driven planning from local obstacle/neighboring robots avoidance [1]–[3]. These approaches largely treat interactions among the robot team as safety constraints rather than opportunities to enhance team-level

performance: robots rarely leverage team-level coordination and shared global information to guide long-horizon decision-making. Another line of work studies formation-preserving or centroid-driven navigation, where coordinated motions yield more globally consistent behaviors, e.g., maintaining formation structure and avoiding oscillations/conflicts [4], [5]. Yet, these approaches typically assume a known environment or fixed/specific formation objectives, which limits their flexibility in open-ended navigation tasks where robots must dynamically adapt their behaviors to evolving map information. Another closely related domain is Multi-Agent Path Finding (MAPF), where the goal is to compute globally coordinated, conflict-free trajectories for all robots (usually offline), often emphasizing scalability in large structured environments [6]–[8]. Despite the global coordination achieved by MAPF approaches, they usually assume a fully known environment and lack the ability to reason about partial observability, exploration, or online map updates.

To address these challenges, we propose *ORION*, a novel deep reinforcement learning-based framework designed for cooperative multi-agent online navigation. ORION first introduces a fused graph-based representation of each agent's prior and online updated maps, enabling the downstream policy network to jointly reason about known structures and newly observed areas. We further propose an option-critic structure in which an *option* denotes a high-level policy that either focuses on the agent's individual navigation or on supporting teammates by collecting team-level informative observations about the environment. Each option persists for multiple time steps and provides the corresponding low-level action guidance. A learned termination mechanism then decides, based on local observations and the team's state, whether to continue the current option or switch to a more suitable one. Embedding this option mechanism into both the policy and critic networks enables ORION to judge/evaluate the long-term impact of option sequencing, allowing robots to adopt behaviors that balance individual progress with team-level gains. Within this option framework, we further design a two-stage cooperation strategy that captures distinct behaviors before and after an agent reaches its target. Before arrival, agents navigate toward their targets while opportunistically gathering information that can reduce future detours and alleviate team-level information gaps. After arrival, instead of remaining idle, agents may choose to explore nearby areas to assist other agents still en route, adaptively deciding how much additional exploration is beneficial without jeopardizing their timeliness. This strategy is validated in our experiments to benefit the makespan by resolving remaining uncertainties for agents still "suffering" from individual exploration-exploitation trade-offs. We evaluate ORION extensively across hundreds of simulated maps, large-scale Gazebo environments, and real-world deployments with varying team sizes and start–target assignments. ORION consistently outperforms state-of-the-art classical and learning-based planners by $6.9\,\%$-$13.4\,\%$ in makespan while maintaining real-time inference efficiency. We further demonstrate up to an additional $14.2\,\%$ improvement arising specifically from our two-stage cooperation strategy. Finally, real-world experiments further demonstrate that ORION transfers without additional training/tuning, highlighting its practicality for cooperative navigation under partial observability.

II. Related Work

A. Navigation in Partially Known Environments

Navigation in fully known environments has been extensively studied, with search-based planners relying on efficient heuristic-guided graph expansion and sampling-based planners exploring the continuous space through incremental sampling [9]–[12]. While these methods are effective in static and fully known settings, they tend to degrade in partially known environments due to their limited ability to anticipate unobserved environment and the high computational cost of frequent replanning.

Learning-based navigation methods, particularly deep reinforcement learning, have shown promise in reasoning over uncertainty without hand-crafted heuristics. Many of these approaches, however, are designed primarily for local/motion-level planning with sensory inputs such as RGB-D images or point clouds [13], [14], making them difficult to scale to large, complex environments where long-horizon reasoning is essential [15]. Several approaches employ belief-informed [16] or hierarchical policies [17] to enhance spatial reasoning and long-term waypoint planning, but still require additional mechanisms to prune redundant exploration or maintain plan quality at scale. Parallel efforts incorporate generative models to predict unknown regions [18], [19] or to guide long-term planning [20], which alleviates myopic behaviors. Overall, although these methods significantly advance navigation under uncertainty, they operate as heuristic, single-agent policies and do not benefit from team-level cooperation. Moreover, their decision-making is tightly coupled to individual belief updates and is not suitable to handle inter-agent interactions, making them difficult to transfer directly to multi-agent navigation settings.

B. Cooperative Multi-Agent Navigation

Beyond single-robot navigation, achieving cooperation across a team of robots poses more challenges. Most works in multi-agent navigation focus on guaranteeing collision-free motion at the kinematic level [1]–[3], often in cluttered/crowded environments. These methods primarily treat other agents as local dynamic constraints, and their policies are designed to ensure short-horizon safety and feasibility. While some of them explicitly consider environmental uncertainty [21], this decision-making is still tightly coupled to individual belief updates, making it difficult to endow robots with long-horizon, team-level reasoning in large, partially known environments. Another line of work studies multi-agent navigation while keeping a free-floating formation of specific shape [4], [5]. These approaches design behavior-based rules to induce globally consistent motion, but they typically assume known environments, fixed formation objectives, or pre-specified connectivity patterns. Multi-Agent Path Finding (MAPF) offers a more global view of coordination, aiming to compute conflict-free trajectories that

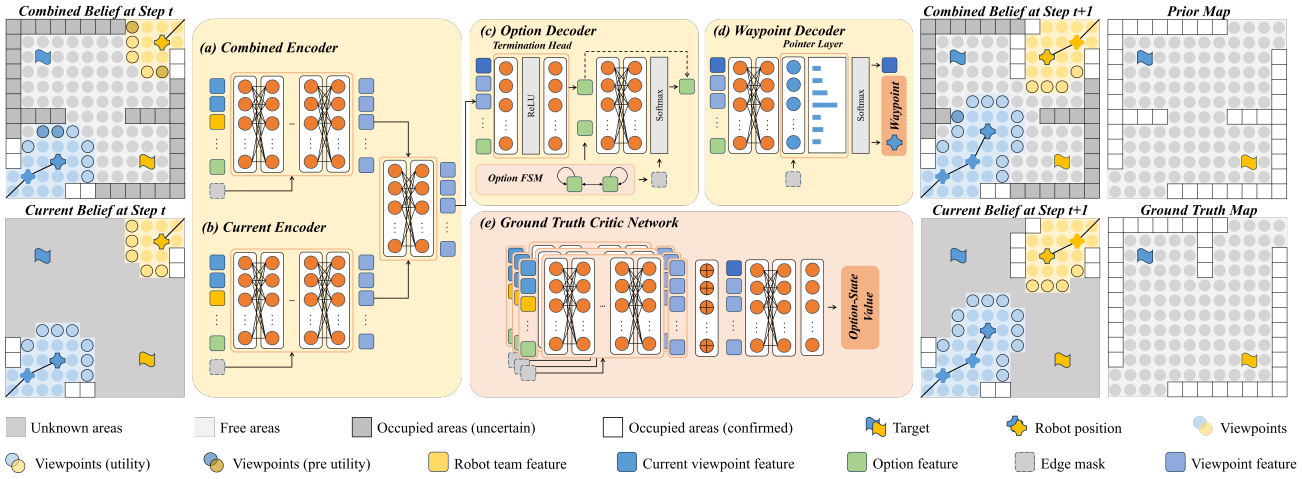

Fig. 2. Policy and multi-agent critic networks of ORION. The combined and current encoders fuse prior information with online observations into joint features. A termination head and option decoder then decide whether to maintain the current option or switch to a new valid one, while the waypoint decoder integrates the option feature with the current node feature to select a waypoint from the agent's neighboring nodes. In parallel, the critic network, conditioned on the states, actions, and selected options of all agents, provides centralized value estimates. During training, it leverages the ground-truth map as privileged information to estimate option–state values that capture the long-term team return.

minimize makespan. Representative methods include bounded-suboptimal [6], prioritized/rule-based search [7], as well as learning-augmented [8] and lifelong variants [22] that scale to thousands of agents. Despite their strong scalability and coordination in discrete, fully known environments, extending MAPF to continuous, partially known environments with online execution remains challenging. A few works move closer to cooperative reasoning by studying multi-agent path topologies without explicit communication [23], enabling targeted rendezvous/message passing among agents [24]–[26], or co-optimizing policies with reconfigurable environments [27]. Yet, they remain largely heuristics-based, relying on implicit communication that emerges from local observations rather than deliberate intent sharing, and thus lack mechanisms to model temporally extended, altruistic cooperation, such as when to sacrifice individual progress or re-engage to assist teammates.

III. The ORION Framework

A. Problem Formulation

We study multi-agent navigation in partially known environments with online map sharing. A team of N agents $\mathcal{A} = \{1, \dots, N\}$ is initially provided a 2D prior occupancy map $\mathcal{M}^-$ consisting of free regions $\mathcal{M}_f^-$ and occupied regions $\mathcal{M}_o^-$. The true environment is described by a ground truth map $\mathcal{M}_g$, which may partially differ from $\mathcal{M}^-$ due to layout changes, such as shelf rearrangements. During execution, agents share sensor observations through perfect global communication to incrementally build a current map $\mathcal{M}$, which is initially unknown and gradually reveals the true structure of the environment. To reason jointly over prior knowledge and newly revealed areas, agents maintain a combined map $\hat{\mathcal{M}}$, initialized as $\mathcal{M}^-$ and continuously updated using observations accumulated in $\mathcal{M}$. The environment is modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where nodes $v \in \mathcal{V}$ are uniformly sampled from free space in $\hat{\mathcal{M}}$ (and $\mathcal{M}$) and edges $(v_i, v_j) \in \mathcal{E}$ represent collision-free transitions between neighboring nodes, with each node connected to up to k neighbors. Each agent starts from an initial node $s_i \in \mathcal{V}$ and aims to reach a designated target node $g_i \in \mathcal{V}$. At each decision step, an agent may move to a neighboring node or remain stationary. Agents must avoid *vertex and edge collisions*, i.e., no two agents may occupy the same node at the same time, i.e., $v_t^i \neq v_t^i$ for all $i \neq j$ or traverse the same edge in opposite directions, i.e., $(v_t^i, v_{t+1}^i) \neq (v_{t+1}^j, v_t^j)$ for all $i \neq j$. The objective is to generate a set of collision-free trajectories $\Psi = \{\psi_1, \dots, \psi_N\}$ that guide all agents to their targets through sequence of neighboring nodes while minimizing the team makespan, defined as $\min_\Psi \max_{i \in \mathcal{A}} T_i$, where T_i denotes the arrival time (or distance) of agent i.

During the task, robots undergo two distinctive stages: During the *pre-arrival* stage, an agent prioritizes navigating toward its target while opportunistically collecting information that may reduce uncertainty for the team. Once it reaches its target, i.e., in the *post-arrival* stage, the agent autonomously decides to either remain at its target until the overall task completes, or temporarily depart to explore nearby uncertain areas and assist teammates still en route, provided it can return in time.

B. Observation as Informative Graph

Each node/viewpoint is augmented with semantic attributes beyond 2D coordinates to obtain richer representations: (1) **Utility** u_j: Following [17], [28], frontier measures as boundaries between free space and unknown areas in the current map $\mathcal{M}$, and the utility u_j denotes the number of such frontiers at node v_j, which are visible, i.e., along a collision-free line of sight. (2) **Prior utility** u_j^p: Similar to frontiers, prior frontiers represent the boundary between confirmed free space (validated by the current map) and uncertain obstacles inferred from the prior map. It combines confirmed observations in $\mathcal{M}$ with prior knowledge in $\mathcal{M}^-$. These prior frontiers

capture the potential information gain suggested by the prior map, offering a long-range heuristic of current unknown areas. Prior utility u_j^p is defined as the number of observable prior frontiers at node v_j. (3) **Visited flag** $\delta_{j,i}$: indicates whether v_j has been visited by agent i. (4) **Verified signal** s_j: marks whether v_j lies within $\mathcal{M}$. (5) **Occupancy indicator** $p_{j,i}$: as a ternary variable, encodes whether v_j is currently occupied by agent i. (6) **Target binary** $t_{j,i}$: specifies whether v_j coincides with agent i's own target or with another agent's target (if any). We further cluster all viewpoints with non-zero utility within a range r_b into a *beacon set* [17], representing candidate regions to navigate. We further construct a planning graph $\bar{G}_t = (\bar{V}_t, \bar{E}_t)$ on $\mathcal{M}$, containing samples from both free and unknown cells, where edges connect collision-free pairs of neighbors. Based on $\mathcal{M}$, we define two guideposts that facilitate coordinated navigation: (7) **Navigation guidepost** g_i: the Dijkstra trajectory on $\bar{G}_t$ from agent i to the beacon nearest the agent's assigned target [29], which serves as a feasible trajectory cue for self-navigation. (8) **Cooperation guidepost** g_i^c: if any other target remains unverified, i.e., not in the explored area, we compute the trajectory from agent i to the beacon nearest the closest unverified target; otherwise $g_i = g_i^c$ if all targets are verified. This guidance makes it feasible for agents to assist teammates.

C. Policy and Critic Networks

*1) **Graph-based Encoders**:* As shown in Figure 2, we design a graph attention-based network that includes an updated and a combined encoder to fuse online, reliable observations from the current map with the rest of the knowledge that may not be up to date from the prior map. The updated encoder processes the current graph G_t into d-dimensional feature $\mathbf{s}^n \in \mathbb{R}^{n\times d}$. Specifically, it begins with a feed-forward layer and then applies multiple masked self-attention layers. The mask set $M \in \mathbb{R}^{n\times n}$, obtained from edge connections E_t, restricts each node's attention to its neighbors. On the other hand, the combined encoder shares the same architecture as the updated encoder but operates on the combined augmented graph $\hat{G}_t$, producing feature $\hat{\mathbf{s}}^n \in \mathbb{R}^{n\times d}$. Subsequently, we fuse the two feature sets $\mathbf{s}^n$ and $\hat{\mathbf{s}}^n$ via a cross-attention layer into the robot state feature $\mathbf{s} \in \mathbb{R}^d$, enabling the model to selectively integrate prior information by down-weighing inconsistent priors and emphasizing those that align with the updated belief.

*2) **Option Decoder** π_ϑ:* During multi-agent navigation, an agent may either navigate toward its designated target or assist others by collecting more information that benefits the team, each demanding distinct and sometimes conflicting reasoning. In such cases, adaptive switching between self-directed and cooperative behaviors becomes crucial. To balance individual efficiency and collective performance for improved overall task success, we introduce the option-critic structure [30], where an option denotes a high-level behavioral mode characterized by a termination head function and an intra-option policy, while the critic provides a value-based estimation guiding high-level option/low-level action optimization. Specifically, we augment the policy network with an *option decoder*, which determines whether to continue the current high-level behavior or terminate and switch to another. Given the current state representation feature $\mathbf{s}_t$ and the active option z_{t-1} at the last step, we define a termination head function $\beta_\vartheta(\mathbf{s}_t, z_{t-1})$ that estimates the likelihood of terminating the current option at step t. Specifically, we first obtain a joint feature representation by combining the current node feature and the embedding of the previous option $h_t = \mathbf{s}_t + e(z_{t-1})$, where $e(z_{t-1}) \in \mathbb{R}^d$ denotes the learnable option embedding. We parameterize the termination function as $\beta_\vartheta(\mathbf{s}_t, z_{t-1}) = \sigma\big(\mathrm{MLP}_\vartheta(h_t)\big)$, where $\mathrm{MLP}_\vartheta(\cdot)$ is a two-layer feed-forward network with ReLU activation, and $\sigma(\cdot)$ denotes the logistic sigmoid. Finally, a Bernoulli sampling $\mathbf{1}_{\text{term}}/\mathbf{1}_{\text{cont}} \sim \mathrm{Bern}(\beta_\vartheta)$ determines whether the current option should terminate or not.

Although our option framework enables flexible switching between high-level behavioral modes, unconstrained transitions may lead to inconsistent/invalid options. Here we introduce a *finite-state machine* (FSM) that encodes admissible transitions between options at different stages of the task. For a general option set $\mathcal{Z} = \{z_1, \ldots, z_K\}$, we define a binary transition matrix $M(\mathbf{s}_t) \in \{0,1\}^{K\times K}$, where each entry $M_{ij}(\mathbf{s}_t)$ specifies whether transitioning from option i to option j is allowed. This mask captures explicit coordination rules at different stages. The option-selection distribution is therefore computed via a masked softmax:

$$\pi_\vartheta(z_t \mid \mathbf{s}_t, z_{t-1}) = \frac{M_{z_{t-1},z_t}(\mathbf{s}_t)\,\exp\big(\beta_\vartheta(\mathbf{s}_t, z_t)\big)}{\sum_{z'\in\mathcal{Z}} M_{z_{t-1},z'}(\mathbf{s}_t)\,\exp\big(\beta_\vartheta(\mathbf{s}_t, z')\big)}. \tag{1}$$

In our navigation setting, the option space simplifies into two high-level modes, i.e., self-directed navigation and cooperative assistance, yet the FSM remains essential: these two modes play fundamentally different roles before and after the agent reaches its individual target. During the *pre-arrival* phase, both behaviors are permissible, enabling flexible coordination. In contrast, during the *post-arrival* phase, our FSM suppresses self-directed continuation and biases option transitions toward the cooperative mode, reflecting the fact that an agent that has completed its own task should primarily support others. At execution time, the termination decision $\mathbf{1}_{\text{term}}/\mathbf{1}_{\text{cont}} \sim \mathrm{Bern}(\beta_\vartheta)$ determines whether the agent remains committed to the current high-level mode or consults the FSM selector to transition into another valid option:

$$z_t = \begin{cases} z_{t-1}, & \text{if } \mathbf{1}_{\text{term}} = 0, \\ z' \sim \pi_\vartheta(\cdot \mid \mathbf{s}_t, z_{t-1}), & \text{if } \mathbf{1}_{\text{term}} = 1. \end{cases}$$

Once a high-level option z_t is selected (or continued from the previous step), the intra-option policy is realized through a pointer-network decoder that generates the next waypoint conditioned on both the current node representation and the option embedding. Specifically, we first form the option-conditioned latent feature $\tilde{\mathbf{h}}_t = \mathbf{h}_t + \mathbf{e}(z_t)$, where $\mathbf{h}_t \in \mathbf{s}_t$ denotes the encoded feature of the current node and $\mathbf{e}(z_t)$ is a learned embedding that modulates the decoder according to the current option. This additive conditioning biases the decoder's attention toward option-specific spatial preferences, allowing the model to express distinct behaviors under different options. The decoder then attends over the neighboring node features

$H_i = \{\mathbf{h}_j\}_{j\in\mathcal{N}(i)}$, where $\mathcal{N}(i)$ indicates the set of node v_i's neighbors, producing an enhanced node representation $\mathbf{h}'_t = \mathrm{Decoder}(\tilde{\mathbf{h}}_t, H_i)$, which integrates both the option-conditioned query and contextual neighbor information via cross-attention. Based on this decoded representation, a pointer layer computes attention scores over all candidate waypoints, and finally, the decoder outputs option-specific action logits $\pi_\theta(a_t \mid \mathbf{s}_t, z_t) = \mathrm{softmax}\big(\mathrm{Pointer}(\mathbf{h}'_t, H_i)\big)$, where the selected waypoint $\hat{a}_t = \arg\max_{a_t} \pi_\theta(a_t \mid \mathbf{s}_t, z_t)$ determines the robot's next move.

*3) **Multi-Agent Critic Networks**:* For agent i, local feature $\mathbf{h}_i$ is extracted from the critic encoder to capture individual observations without inter-agent interaction. Enhanced feature $\tilde{\mathbf{h}}_i$ further aggregates team-wide context via cross-attention layers. To model coordination, $\tilde{\mathbf{h}}_{i,j}$ denotes the relational embedding between agent i and j, capturing their relative states under a shared team belief. These features are concatenated to form the local state–action embedding $\mathbf{u}_{i,j} = f_{\mathrm{emb}}([\tilde{\mathbf{h}}_i, \tilde{\mathbf{h}}_{i,j}])$, which jointly integrates the agent's local state, inter-agent relationships, and the global context of the environment. Following MAAC [31], the critic computes a query $\mathbf{q}_i = W_q\mathbf{h}_i$ and attends to keys and values $\mathbf{k}_j = W_k\mathbf{u}_{i,j}$, $\mathbf{v}_j = W_v\mathbf{u}_{i,j}$ via

$$\alpha_{ij} = \frac{\exp(\mathbf{q}_i^\top \mathbf{k}_j)}{\sum_{\ell\neq i}\exp(\mathbf{q}_i^\top \mathbf{k}_\ell)}, \qquad \mathbf{c}_i = \sum_{j\neq i}\alpha_{ij}\mathbf{v}_j.$$

Finally, the critic forms $Q_i = f_Q([\mathbf{u}_{i,j}]_{j\in\mathcal{N}})$, where f_Q is the embedding layer to output the centralized action-value estimation Q_i for agent i in the robot team $\mathcal{N}$, which integrates local features, multi-agent interactions, and action-specific neighborhood structure. Our critic network adopts the same graph-transformer backbone as the policy network, but differs in its use of a *privileged graph encoder*. During training, this encoder has access to privileged ground-truth information (i.e., the ground-truth map), enabling the critic to provide low-variance estimations on long-term returns to the decentralized policies. Unlike the actor's dual encoders (one for the combined graph and one for the current graph), the critic employs a single unified encoder that captures the complete relational context of all agents and their robot beliefs. Besides the state representation input, the critic also takes the option feature selected at the previous step as input, which benefits both option and waypoint optimization during critic learning [31]. The value estimate is explicitly conditioned on the agent's previous option z_{t-1}, representing the high-level mode active at the current step. Specifically, the option embedding is added to the current node feature:

$$\mathbf{h}_t^i = \mathrm{Gather}(H_t, \mathrm{index} = i) + e(z_{t-1}^i), \tag{2}$$

where H_t is the encoded graph representation and $e(\cdot)$ is an embedding layer. The resulting feature $\mathbf{h}_t^i$ is then refined by a transformer decoder queried with other agents' features, forming an option-aware latent representation. Finally, instead of outputting a waypoint distribution, the critic decoder produces the *option-state values* as the long-term estimation.

Our critic network enables termination learning by contrasting/advantaging the expected return of continuing the current option with that of terminating it. For agent i, given the per-agent Q-values $Q_\vartheta^i(\mathbf{s}_t, a_t, z_t)$ and the counterfactual value if the option terminates $Q_\vartheta^i(\mathbf{s}_t, a_t, \bar{z}_t)$, we compute the advantage of termination as $A^i_{\text{term}} = Q_\vartheta^i(\mathbf{s}_t, a_t, \bar{z}_t) - Q_\vartheta^i(\mathbf{s}_t, a_t, z_t)$, which quantifies whether switching options yields a higher expected value than continuing. In the policy network's option decoder, the termination head outputs $\mathbf{1}_{\text{term}}/\mathbf{1}_{\text{cont}} \sim \mathrm{Bern}(\beta_\vartheta)$ through a scalar logit, from which we define the Bernoulli probabilities of termination and continuation: $p^i_{\text{term}} \sim \log\beta_\vartheta(\cdot \mid \mathbf{s}_t, z_{t-1})$ and $p^i_{\text{cont}} \sim \log\beta_\vartheta(\cdot \mid \mathbf{s}_t, \bar{z}_{t-1})$. Given the binary termination signal $\delta_t^i \in \{0, 1\}$ from the termination head, the likelihood of the observed decision is $p^i_\vartheta = \mathbf{1}_{\text{term}}p^i_{\text{term}} + \mathbf{1}_{\text{cont}}p^i_{\text{cont}}$.

Overall, we optimize the termination head by maximizing the expected return differential, i.e., agents are encouraged to terminate when doing so leads to higher option-state values. The termination head loss is thus defined as $\mathcal{L}_\vartheta = -\mathbb{E}\big[\log p^i_\vartheta\, A^i_{\text{term}}\big]$. On the other hand, following [32], the policy loss is further improved by our option-regularized policy network:

$$\mathcal{L}_\pi = \mathbb{E}_{(\mathbf{s}_t, z_t)}\Bigg[\sum_{a_t}\pi_\theta(a_t \mid \mathbf{s}_t, z_t)\Big(\alpha\,\log\pi_\theta(a_t \mid \mathbf{s}_t, z_t) - Q_\vartheta(\mathbf{s}_t, a_t, z_t)\Big)\Bigg]. \tag{3}$$

Finally, the overall loss is $\mathcal{L}_{\text{policy}} = \mathcal{L}_\pi + \lambda_\vartheta\,\mathcal{L}_\vartheta$, where λ_ϑ balances termination learning. Following [32], the critic is optimized via a temporal-difference regression objective.

IV. Experiments

A. Comparisons in Simulated Maps

We expand the navigation dataset released in [16] to train ORION with a sensor range of $20\,\mathrm{m}$. For graph construction, we uniformly sample 600 points across the environment and treat points in known free space as candidate viewpoints. Each viewpoint is connected to its 20 nearest neighbors while retaining only collision-free edges. ORION is trained on a workstation with one i9-10980XE CPU and two NVIDIA GeForce RTX 3090 GPUs. Training typically converges after $60\,000$ episodes and takes around one day. Following recent navigation and exploration benchmarks [16], [25], we further introduce a benchmark of 500 large and challenging maze-like simulated maps to evaluate ORION against representative baselines. For the learning-based baseline, we consider **MAContext** [16], a multi-agent extension of a DRL navigation planner designed for partially known environments. For classical MAPF solvers, we include 1) **LNS2** [7], a highly efficient and scalable planner that iteratively repairs collision-containing path sets by replanning for a subset of colliding agents, and 2) **EECBS** [6], a bounded-suboptimal MAPF planner that combines high-level Explicit Estimation Search with low-level focal search. To deploy LNS2 and EECBS in our partially known setting, we align their execution with ORION by using the same graph-construction pipeline: At each decision timestep, they replan on the constructed graph and select the first waypoint of the planned path as the next waypoint, with the same replanning frequency and collision definitions as ORION. To further analyze our model, we additionally evaluate four ablation variants: 1) **ORION**

TABLE I
COMPARISONS ON $160\,\mathrm{m}\times150\,\mathrm{m}$ SIMULATED MAPS. EACH ENTRY REPORTS THE AVERAGE VALUE WITH ITS VARIANCE IN PARENTHESES.

Planner	3 Agents		4 Agents		5 Agents		10 Agents	
	Max Dist (m)↓	**Steps** ↓	**Max Dist** (m)↓	**Steps** ↓	**Max Dist** (m)↓	**Steps** ↓	**Max Dist** (m)↓	**Steps** ↓
MAContext w/o dual-stage [16]	522.07 (±162.09)	64.59	510.11 (±170.3)	62.64	495.39 (±151.08)	61.03	583.66 (±177.65)	71.52
MAContext [16]	503.3 (±152.98)	61.63	495.45 (±129.99)	61	478.49 (±110.82)	59.16	500.77 (±168.78)	60.53
EECBS [6]	526.12 (±151.22)	66.08	501.78 (±127.77)	62.64	477.12 (±129.86)	59.49	468.3 (±136.55)	58.6
LNS2 [7]	520.99 (±141.15)	65.09	499.24 (±119.46)	62.52	474.32 (±127.82)	59.49	466.24 (±141.68)	58.44
ORION w/o option	492.83 (±139.02)	59.98	484.73 (±120.26)	58.98	467 (±119.51)	56.92	457.29 (±129.14)	55.56
ORION w/o current	490.13 (±135.07)	60.41	470.53 (±93.45)	58.42	467.88 (±98.65)	58.54	445.71 (±107.15)	55.32
ORION w/o combined	500.11 (±137.87)	61.45	480.81 (±138.03)	59.98	472.29 (±133.14)	58.76	468.28 (±112.92)	57.6
ORION w/o dual-stage	462.91 (±118.24)	56.08	453.85 (±131.57)	55.2	447.7 (±125.5)	54.54	438.31 (±141.91)	54.69
ORION	**455.83** (±108.51)	**55.36**	**441.99** (±107.2)	**53.86**	**439.24** (±111.08)	**53.24**	**435.76** (±99.28)	**53.26**

TABLE II
COMPARISONS ON GAZEBO SIMULATIONS. EACH ENTRY REPORTS THE MAXIMUM, AVERAGE, AND MINIMUM TRAVEL DISTANCE.

Planner	3 Agents			4 Agents			5 Agents		
	Max Dist ↓	**Avg Dist** ↓	**Min Dist** ↓	**Max Dist** ↓	**Avg Dist** ↓	**Min Dist** ↓	**Max Dist** ↓	**Avg Dist** ↓	**Min Dist** ↓
MAContext [16]	178.83	110.95	**54.55**	228.13	120.53	57.78	141.98	91.5	**60.75**
ORION w/o option	111.74	96.6	71.45	102.46	81.86	57.02	108.7	92.26	64.49
ORION	**75.99**	**64.2**	55.34	**72.68**	**63.5**	**54.14**	**95.64**	**75.06**	64.58

w/o option, removing our option-critic structure, 2) **ORION w/o current**, current map being unavailable, 3) **ORION w/o combined**, excluding the combined map, and 4) **ORION w/o dual-stage**, disabling our pre-/post-arrival stages. These variants help isolate the role of our proposed option-critic's hierarchical decision-making, dual map representations, and dual-stage cooperation strategy.

As shown in Table I, ORION achieves the best overall performance across all team sizes, reducing makespan by up to $20\,\%$ over state-of-the-art planners. Traditional baselines (LNS2 and EECBS) maintain competitive as the team grows; however, ORION consistently outperforms them, exhibiting a 12.5–$13.4\,\%$ advantage at 3 agents and still retaining 6.5–$6.9\,\%$ improvement at 10 agents. In contrast, the learning-based baseline MAContext degrades with scale, with makespan increasing from $478.5\,\mathrm{m}$ to $500.8\,\mathrm{m}$ as the number of agents increases from 5 to 10. ORION, however, maintains stable makespan and decision steps, highlighting stronger scalability and generalization. Although our task objective focuses on minimizing the team's makespan, ORION also improves the average and minimum travel distance (Fig. 3), further indicating better overall planning efficiency. Qualitatively, ORION exhibits targeted, timing-aware cooperation: agents with low-cost detours or completed goals often help resolve key uncertainties for bottleneck teammates rather than assisting uniformly across the team. Two ablations further indicate that both maps are necessary. ORION w/o combined, i.e., without prior information, causes a clear makespan drop (7.5–$9.7\,\%$ at 3–5 agents), since agents spend more effort exploring before exploiting prior observations. Removing the current map (ORION w/o current) also degrades performance across all team sizes, showing that up-to-date observations are equally important for correcting prior-map errors. Together, these results confirm that the combined map provides useful environmental priors, while the current map supplies accurate online updates.

We further apply our dual-stage cooperation strategy to both ORION and MAContext by allowing agents that have already reached their targets to continue exploring nearby uncertain areas. This post-arrival assistance consistently improves team performance, with benefits becoming more pronounced as team size increases. For instance, MAContext improves makespan from $583.7\,\mathrm{m}$ to $500.8\,\mathrm{m}$ when equipped with the dual-stage strategy, highlighting the importance of leveraging idle agents to resolve remaining team-level uncertainties.

B. Validation in Large-Scale Gazebo Simulations

We further evaluate our approach in a $70\,\mathrm{m}\times60\,\mathrm{m}$ Gazebo warehouse environment, where shelf rearrangements cause partial discrepancies between the prior map and the ground truth, thereby simulating realistic online navigation under map changes. ORION and MAContext are deployed on ground vehicle teams (maximum speed $2\,\mathrm{m/s}$) equipped with Velodyne VLP16 LiDARs. As shown in Table II, we compare ORION, ORION w/o option-critic, and MAContext with teams of 3, 4, and 5 robots. ORION consistently achieves the best performance in terms of maximum and average distance. We also observe that, with the option module enabled, ORION brings a performance improvement of 12–$32\,\%$ in maximum distance and 19–$34\,\%$ in average distance, which further highlights the importance of the option module. While ORION exhibits a slightly larger minimum distance for the largest team, this reflects deliberate post-arrival exploration. Agents that reach their targets early, or are less affected by map changes, take short detours to reduce uncertainty for slower teammates. This trade-off slightly increases individual distance but provides larger team-level gains. It emerges naturally from the option-critic formulation and team-level objective rather than hard-coded heuristics.

Fig. 4 illustrates a representative cooperative behavior enabled by ORION: agent 1 reaches its target first (red trajectory) and then moves toward the vicinity of target 2 to share

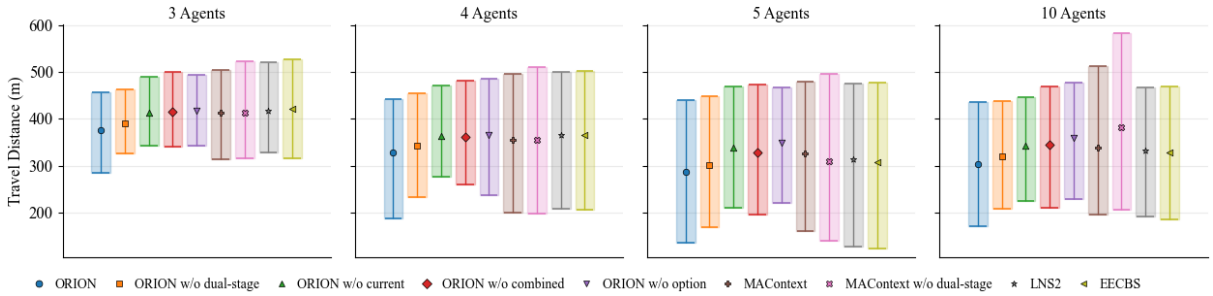

Fig. 3. Comparison of travel distances on simulated maps. For each planner, each bar shows the makespan, average travel distance, and minimum team distance as the top, middle, and bottom markers, respectively. Results are reported for teams of 3, 4, 5, and 10 agents.

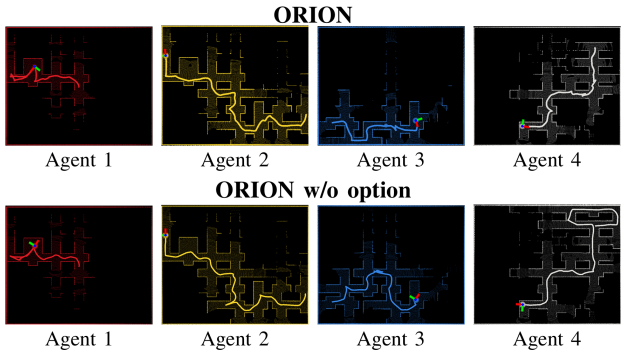

Fig. 4. Gazebo Experiments. ORION yields more efficient decentralized coordination than ORION w/o option, as shown by the local maps of the four agents.

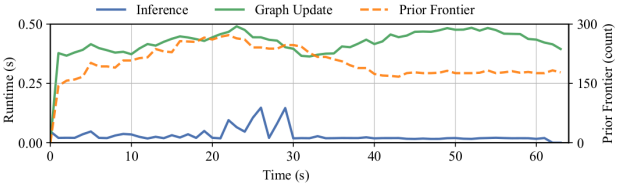

Fig. 5. Runtime performance in Gazebo simulation. ORION maintains real-time performance on graph updates and network inference throughout execution, while the prior frontier curve reflects how ORION incrementally verifies and corrects uncertain regions in the prior map during online navigation.

informative observations with agent 2. Upon completing this assistance, agent 1 returns to its own target instead of performing unnecessary exploration, highlighting adaptive cooperation enabled by the option-critic framework and the proposed dual-stage strategy. As shown in Fig. 5, ORION maintains real-time performance throughout the experiments: graph update remains below $0.5\,\mathrm{s}$, and network inference requires less than $0.2\,\mathrm{s}$ (typically under $0.1\,\mathrm{s}$), despite the entire system being implemented in Python. We also report the evolution of prior utility, defined as the number of sampled nodes along the boundary between confirmed free space and uncertain obstacles. The prior utility stabilizes after approximately $40\,\mathrm{s}$, indicating that the team has obtained a sufficiently confident map and no longer needs extra exploration.

C. Real-World Experiments

Finally, we deploy ORION on two ground vehicles (maximum speed $0.5\,\mathrm{m/s}$) equipped with Livox Mid-360 3D LiDARs for odometry and mapping, operating in a $30\,\mathrm{m}\times10\,\mathrm{m}$ office environment. We use OctoMap with an occupancy resolution of $0.4\,\mathrm{m}$ and a waypoint resolution of $0.5\,\mathrm{m}$ to construct the online map. During online execution, all robots share their locally built current maps, allowing the system to periodically merge them into a global current map. The unknown regions in this global current map are filled with prior-map information, producing the combined map used by all robots. As shown in Fig. 6, we evaluate ORION under two different start–target configurations that showcase distinct cooperative behaviors enabled by our framework. In both settings, the team travels approximately $20\,\mathrm{m}$ within $90\,\mathrm{s}$. In case 1, agent 1 (red) benefits from areas previously explored by agent 2 (yellow), demonstrating *passive map sharing*, where information gathered by one agent directly assists another without explicit coordination. In case 2, agent 1 exhibits *post-arrival cooperation*: after reaching its target, it moves toward the vicinity of target 2 to provide updated observations. Once target 2 enters the shared belief map, agent 2 immediately returns to its own target without redundant exploration, maintaining a compact travel distance while acquiring critical, up-to-date environmental information. Together, these real-world results highlight ORION's ability to coordinate through both passive map sharing and active post-arrival assistance, underscoring its practicality for multi-robot deployment.

V. Conclusion

In this paper, we presented ORION, a deep reinforcement learning framework for cooperative multi-agent online navigation in partially changed environments. By fusing prior maps with online perception through a shared graph encoder and leveraging a novel option-critic structure, ORION enables agents to switch adaptively between target-directed navigation and cooperative assistance in a decentralized way. Our dual-stage cooperation strategy further enhances team-level performance by allowing arrived agents to contribute additional information that helps teammates reach their targets more efficiently. Extensive experiments on large-scale maze-like maps, high-fidelity Gazebo simulations, and physical robot deployments demonstrate that ORION achieves robust, real-time cooperation and consistently outperforms state-of-the-art baselines across varying team sizes. A current limitation lies in its assumption of reliable map sharing among agents. In future work, we plan to explicitly investigate more realistic settings with noisy communication/imperfect localization, where

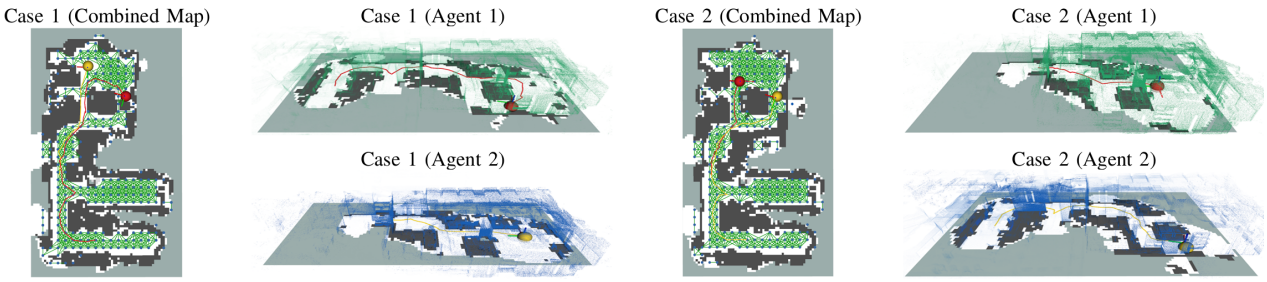

Fig. 6. Real-world experiments under two start–target settings. The left column shows the combined map and trajectories, while the right column shows each agent's current belief and executed path.

agents must reason about uncertain teammates' beliefs and establish adaptive map-sharing and coordination strategies.

REFERENCES

[1] Y. Jin, S. Wei, J. Yuan, and X. Zhang, "Hierarchical and stable multiagent reinforcement learning for cooperative navigation control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 1, pp. 90–103, 2021.

[2] C. Yu, H. Yu, and S. Gao, "Learning control admissibility models with graph neural networks for multi-agent navigation," in *Conference on robot learning*. PMLR, 2023, pp. 934–945.

[3] B. Wang, Z. Liu, Q. Li, and A. Prorok, "Mobile robot path planning in dynamic environments through globally guided reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6932–6939, 2020.

[4] H. G. Tanner and A. Boddu, "Multiagent navigation functions revisited," *IEEE Transactions on Robotics*, vol. 28, no. 6, pp. 1346–1359, 2012.

[5] L. Quan et al., "Robust and efficient trajectory planning for formation flight in dense environments," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4785–4804, 2023.

[6] J. Li, W. Ruml, and S. Koenig, "Eecbs: A bounded-suboptimal search for multi-agent path finding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 14, 2021, pp. 12 353–12 362.

[7] J. Li, Z. Chen, D. Harabor, P. J. Stuckey, and S. Koenig, "Mapf-lns2: Fast repairing for multi-agent path finding via large neighborhood search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 9, 2022, pp. 10 256–10 265.

[8] G. Sartoretti et al., "Primal: Pathfinding via reinforcement and imitation multi-agent learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2378–2385, 2019.

[9] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain," *IEEE transactions on robotics*, vol. 21, no. 3, pp. 354–363, 2005.

[10] M. Likhachev, D. I. Ferguson, G. J. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic a*: An anytime, replanning algorithm." in *ICAPS*, vol. 5, 2005, pp. 262–271.

[11] 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.

[12] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Batch informed trees (bit*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 3067–3074.

[13] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *2017 ieee international conference on robotics and automation (icra)*. IEEE, 2017, pp. 1527–1533.

[14] J. Jin, N. M. Nguyen, N. Sakib, D. Graves, H. Yao, and M. Jagersand, "Mapless navigation among dynamics with social-safety-awareness: a reinforcement learning approach from 2d laser scans," in *2020 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2020, pp. 6979–6985.

[15] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 31–36.

[16] J. Liang, Z. Wang, Y. Cao, J. Chiun, M. Zhang, and G. A. Sartoretti, "Context-aware deep reinforcement learning for autonomous robotic navigation in unknown area," in *Conference on Robot Learning*. PMLR, 2023, pp. 1425–1436.

[17] J. Liang, Y. Cao, Y. Ma, H. Zhao, and G. Sartoretti, "Hdplanner: Advancing autonomous deployments in unknown environments through hierarchical decision networks," *IEEE Robotics and Automation Letters*, vol. 10, no. 1, pp. 256–263, 2024.

[18] C. Ho et al., "Mapex: Indoor structure exploration with probabilistic information gain from global map predictions," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 13 074–13 080.

[19] Y. Wang, H. He, J. Liang, Y. Cao, R. Chakraborty, and G. A. Sartoretti, "Cogniplan: Uncertainty-guided path planning with conditional generative layout prediction," in *Conference on Robot Learning*. PMLR, 2025, pp. 1382–1396.

[20] Y. Cao, J. Lew, J. Liang, J. Cheng, and G. Sartoretti, "Dare: Diffusion policy for autonomous robot exploration," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 11 987–11 993.

[21] G. Wagner and H. Choset, "Path planning for multiple agents under uncertainty," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 27, 2017, pp. 577–585.

[22] H. Jiang, Y. Wang, R. Veerapaneni, T. Duhan, G. Sartoretti, and J. Li, "Deploying ten thousand robots: Scalable imitation learning for lifelong multi-agent path finding," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 1–7.

[23] C. I. Mavrogiannis and R. A. Knepper, "Multi-agent path topology in support of socially competent navigation planning," *The International Journal of Robotics Research*, vol. 38, no. 2-3, pp. 338–356, 2019.

[24] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau, "Tarmac: Targeted multi-agent communication," in *International Conference on machine learning*. PMLR, 2019, pp. 1538–1546.

[25] D. M. S. Tan, Y. Ma, J. Liang, Y. C. Chng, Y. Cao, and G. Sartoretti, "Ir 2: Implicit rendezvous for robotic exploration teams under sparse intermittent connectivity," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 13 245–13 252.

[26] S. Agarwal, R. Muthukrishnan, W. Gosrich, V. Kumar, and A. Ribeiro, "Lpac: Learnable perception-action-communication loops with applications to coverage control," *IEEE Transactions on Robotics*, 2025.

[27] Z. Gao, G. Yang, and A. Prorok, "Co-optimizing reconfigurable environments and policies for decentralized multi-agent navigation," *IEEE Transactions on Robotics*, 2025.

[28] B. Yamauchi, "Frontier-based exploration using multiple robots," in *Proceedings of the second international conference on Autonomous agents*, 1998, pp. 47–53.

[29] E. W. Dijkstra, "A note on two problems in connexion with graphs," in *Edsger Wybe Dijkstra: his life, work, and legacy*, 2022, pp. 287–290.

[30] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[31] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2019, pp. 2961–2970.

[32] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. Pmlr, 2018, pp. 1861–1870.