Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

iHERO: Interactive Human-oriented Exploration and Supervision Under Scarce Communication

Zhuoli Tian, Yuyang Zhang, Jinsheng Wei and Meng Guo*
College of Engineering, Peking University

***Abstract*—Exploration of unknown scenes before human entry is essential for safety and efficiency in numerous scenarios, e.g., subterranean exploration, reconnaissance, search and rescue missions. Fleets of autonomous robots are particularly suitable for this task, via concurrent exploration, multi-sensory perception and autonomous navigation. Communication however among the robots can be severely restricted to only close-range exchange via ad-hoc networks. Although some recent works have addressed the problem of collaborative exploration under restricted communication, the crucial role of the human operator has been mostly neglected. Indeed, the operator may: (i) require timely update regarding the exploration progress and fleet status; (ii) prioritize certain regions; and (iii) dynamically move within the explored area; To facilitate these requests, this work proposes an interactive human-oriented online coordination framework for collaborative exploration and supervision under scarce communication (iHERO). The robots switch smoothly and optimally among fast exploration, intermittent exchange of map and sensory data, and return to the operator for status update. It is ensured that these requests are fulfilled online interactively with a pre-specified latency. Extensive large-scale human-in-the-loop simulations and hardware experiments are performed over numerous challenging scenes, which signify its performance such as explored area and efficiency, and validate its potential applicability to real-world scenarios. The videos are available on https://zl-tian.github.io/iHERO/.**

I. Introduction

Exploration of unknown and hazardous scenes before allowing humans inside is crucial for their safety, which can also improve the performance of subsequent tasks with an accurate model. Fleets of UAVs and UGVs have been deployed as extended eyes and arms, e.g., to explore planetary caves in Petráček et al. [20], Klaesson et al. [14]; search and rescue after earthquakes in Couceiro [5]. Many collaborative exploration strategies have been proposed along with the advances in autonomous navigation and perception, e.g., Yamauchi [26], Hussein et al. [12], Colares and Chaimowicz [4], Zhou et al. [28], Patil et al. [18], Guo and Dimarogonas [8]. However, they often assume an all-to-all communication among the robots, where any map observed locally by one robot is immediately available to other robots. This could be impractical in many aforementioned scenes where the communication facilities are unavailable or severely degraded. In such cases, the robots can only exchange information via ad-hoc networks in close proximity. This imposes great challenges on the fleet coordination as communication events and exploration tasks are now closely dependent, thus should be planned simultaneously.

*Corresponding author: Meng Guo, meng.guo@pku.edu.cn.

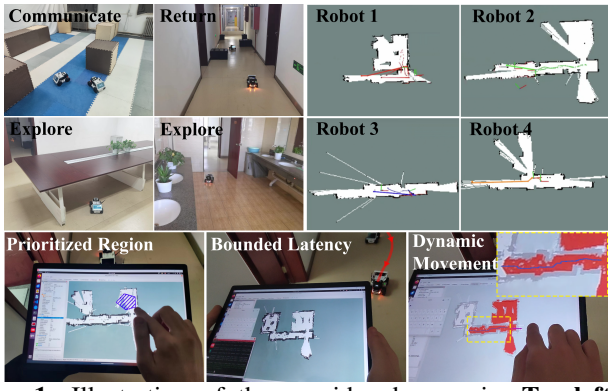

Fig. 1: Illustration of the considered scenario. **Top-left**: 4 UGVs switch among exploration, intermittent communication and return to the operator; **Top-right**: Different local maps at each robot, which are updated via pairwise communication; **Bottom**: three types of human requests: (i) Prioritized regions (in blue) to explore (**Left**); (ii) Status update with bounded latency via return events (**Middle**); (iii) Dynamic movement of the operator within the allowed region (in red) (**Right**).

Furthermore, the role of the human operator and the online interaction with the fleet during exploration are less studied and mostly replaced by a static base for visualization. However, under scarce communication, the operator might be completely *unaware* of the current progress of exploration if none of the robots return to the operator to relay such information. Consequently, the operator can not supervise the exploration task, e.g., checking the system status such as battery and liveness of each robot; or dynamically adjusting the priorities of the areas to be explored, as shown in Fig. 1. Lastly, it is common that the operator might request to move dynamically inside the explored area, in which case to maintain the interactions during movement is quite challenging without a global communication.

A. Related Work

The problem of collaborative exploration has a long history in robotics. The frontier-based method from Yamauchi [26] introduces an intuitive yet powerful metric for guiding the exploration. Originally proposed for a single robot, it has been adapted to multi-robot teams by assigning these frontiers to different robots for concurrent exploration in different ways, e.g., greedily to the nearest robot by distance in Yamauchi [27], distributed and sequential auction in Hussein et al. [12], or the multi-vehicle routing for the optimal sequences

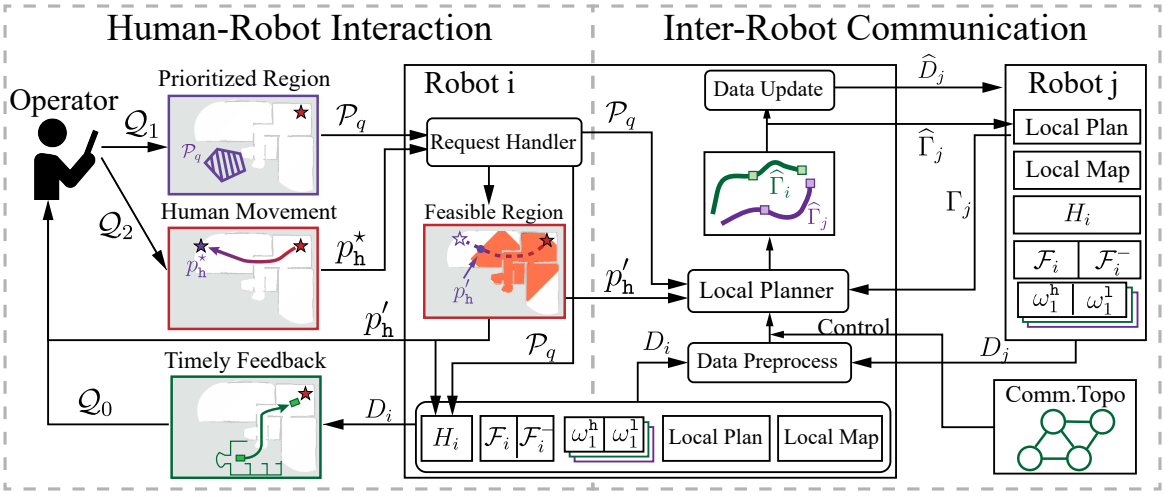

Fig. 2: Illustration of the proposed framework (iHERO), which consists of the inter-robot and robot-operator communication protocol, the collaborative exploration strategy, and more importantly, the online adaptation module to operator requests.

in Zhou et al. [28], or based on the expected information gain in Colares and Chaimowicz [4], Patil et al. [18], Burgard et al. [2]. However, all the aforementioned work assume that all robots can exchange information via wireless communication instantly at all time. Thus, all robots always have access to *the same global map* and the same set of frontiers. This is often impractical or infeasible without a wireless network already installed, especially for subterranean or indoor structures, where the inter-robot communication is poor and limited in range. Consequently, these approaches are not suitable anymore for communication-constrained unknown scenes.

To overcome this challenge of limited communication, many recent work can be found that combines the planning of inter-robot communication and autonomous exploration. The work in Klaesson et al. [14] reports the systematic solution in the DARPA subterranean challenge where an integer linear program (ILP) is formulated over the mobility-communication network. Different intermittent communication strategies have been proposed for different performance metrics, e.g., the "Zonal and Snearkernet" relay algorithm in Vaquero et al. [25], the four-state strategy in Cesare et al. [3], the distributed assignment of relay tasks for static goals in Marchukov and Montano [16], and a centralized integer program to optimize rendezvous points in Gao et al. [7]. On the other hand, fully-connected networks at all time are enforced in Rooker and Birk [21] and further in Pei et al. [19] with a lower-bounded bandwidth. Droppable radios are utilized in Saboia et al. [22] as extended communication relays between robots and a static base station. Their common objective is to maximize the exploration efficiency, however without considering different online interactions with the operator.

Indeed, the human operator plays *an indispensable role* for the operation of robotic fleets, despite their autonomy. In many aforementioned scenarios, the operator should not only be aware of their operation status online, but also directly supervise certain procedures whenever necessary, e.g., prioritized regions to explore, or desired new position of the operator. Almost all related work neglect this aspect and assume a static base station, see, e.g., Klaesson et al. [14], Vaquero et al. [25], Marchukov and Montano [16], Gao et al. [7], Pei et al. [19], Saboia et al. [22]. This interplay brings numerous challenges to be addressed, i.e., how to ensure a timely exchange of information between the operator and the fleet; how to accommodate different requests; and how to cope with the dynamic movement of the operator.

B. Our Method

This work proposes an interactive human-oriented online planning framework for the collaborative exploration of unknown scenes (iHERO), via a fleet of mobile robots under scarce communication. As illustrated in Fig. 2, explicit interactive requests from the operator are formulated including timely status update, prioritized region during exploration, and dynamic movements. The proposed solution is a fully distributed coordination strategy for both collaborative exploration and intermittent communication. It guarantees an upper bound on the maximum latency of the information gathered by any robot to the operator at all time. Moreover, it is fully adaptive to the online requests by adjusting the exploration and communication events accordingly, subject to local communication and tight latency constraints. Lastly, it provides real-time feedback to the operator regarding the allowed region of movement. Extensive human-in-the-loop simulations and hardware experiments are conducted for robotic fleets within office scenes and subterranean caves.

Main contribution of this work is threefold: (i) the novel problem formulation of human-oriented collaborative exploration and supervision under scarce communication, supporting three types of requests. To the best of our knowledge, this problem *has not been addressed* in related work; (ii) the distributed coordination strategy for both fast exploration and inter-robot-operator communication, while ensuring a timely response to online requests; (iii) the numerical validation from extensive simulation and hardware experiments for its

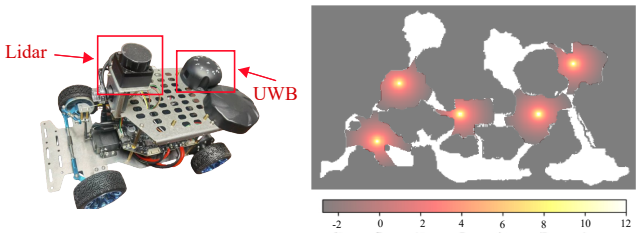

Fig. 3: Left: UGV equipped with Lidar and UWB-based communication unit; **Right**: the communication quality when 5 robots are scattered within a subterranean cave.

applicability to real-world scenarios.

II. Problem Description

A. Workspace and Robots

Consider a 2D workspace $\mathcal{A} \subset \mathbb{R}^2$, of which its *map* including the boundary, freespace and obstacles are all unknown. To obtain a complete map of $\mathcal{A}$, a human operator deploys a team of robots such as UAVs and UGVs around the location $p_0 \in \mathcal{A}$, denoted by $\mathcal{N} \triangleq \{1, \cdots, N\}$. Each robot $i \in \mathcal{N}$ is equipped with various sensors such as IMU, Lidar and RGBD cameras; and actuators to move around the workspace. Moreover, each robot is assumed to be capable of simultaneous localization and mapping (SLAM) with collision avoidance, i.e.,

$$(M_i^+,\, p_i^+,\, \mathbf{p}_g) \triangleq \texttt{SLAM}(p_i,\, p_g,\, M_i,\, \mathcal{A}), \tag{1}$$

where $\texttt{SLAM}(\cdot)$ is a generic SLAM module; $M_i(t) \subset \mathcal{A}$ is the *local* map of robot $i \in \mathcal{N}$ at time $t \geqslant 0$; $p_i(t) \in M_i$ is the 2D pose of robot i; $p_g \in M_i$ is a suitable goal pose within M_i; $\mathbf{p}_g \subset M_i$ is the collision-free path from $p_i(t)$ to p_g within M_i; $M_i^+(t') \subset \mathcal{A}$ is the updated map after traversing the path at time $t' > t$; and $p_i^+(t')$ is the updated pose at time t'.

Remark 1. The exact representation of the map depends on the SLAM algorithm, e.g., the occupancy grid map from Thrun [23], the octomap from Hornung et al. [11], or even the metric-semantic map in Tian et al. [24]. ■

B. Inter-robot Communication

Each robot is equipped with a wireless communication unit for inter-robot information exchange, e.g., WiFi and ultra-wideband (UWB). In particular, robot $i \in \mathcal{N}$ can communicate with robot $j \in \mathcal{N}$ at time $t > 0$, if the communication quality (such as SNR) is above a certain threshold, i.e.,

$$(D_i^+, D_j^+) \triangleq \texttt{Com}\big(D_i, D_j\big), \;\; \text{if } \texttt{Qual}(p_i, p_j, \mathcal{A}) > \underline{\delta}; \tag{2}$$

where $\texttt{Com}(\cdot)$ is the communication module of robots i and j; $D_i(t) \in \mathcal{D}$ is the local data stored at robot i at time $t \geqslant 0$, similar for robot j; $\mathcal{D}$ is a pre-defined data structure, which may include the map data, video/audio stream, coordination plans, requests from the human operator as explained in the sequel; $\texttt{Qual}(\cdot)$ evaluates the communication quality between robots i, j given their positions and the workspace layout; $\underline{\delta} > 0$ is a pre-specified lower bound on the quality; the outputs $D_i^+(t), D_j^+(t)$ are the updated data after communication. By default, $(M_i,\, p_i) \in D_i$ holds for all robots. An illustration of the UWB-based communication is shown in Fig. 3, which is subject to both range and line-of-sight (LOS) constraints.

C. Human-robot Interaction and Online Requests

Similarly, the human operator can receive data from any robot via the same communication module in (2). Namely, the local data and map stored at the operator are denoted by $D_{\text{h}}(t), M_{\text{h}}(t)$, respectively. More importantly, the operator has the following **three** types of requests for the fleet: (i) the operator should obtain the local maps of all robots with a latency smaller than a given threshold $T_{\text{h}} > 0$ at all time, i.e.,

$$\textstyle\bigcup_i M_i(t) \subset M_{\text{h}}(t + T_{\text{h}}),\;\; \forall t \geqslant 0; \tag{3}$$

which means that the explored map by any robot at time t should be known to the operator latest by time $t + T_{\text{h}}$. This request is denoted by Q_0, which is activated at all time; (ii) the locations that should be prioritized during exploration, denoted by $Q_1 \subset \mathcal{A}$. It is fulfilled once the area in Q_1 is explored; (iii) the desired next position of the operator, denoted by $Q_2 \triangleq p_{\text{h}}^\star \in \mathcal{A}$, which is fulfilled after the operator reaches it. As described in the sequel, the Q_2 request might not be feasible due to the constraint (3). Thus, intermediate waypoints that are feasible would be provided for the operator to choose as an interactive process.

Thus, the behavior of each robot is fully determined by its timed sequence of navigation and communication events, i.e.,

$$\Gamma_i \triangleq c_i^0\, \mathbf{p}_i^0\, c_i^1\, \mathbf{p}_i^1\, c_i^2\, \cdots, \tag{4}$$

where c_i^k is a communication event defined by the communication location, the other robot or the operator, and the exchanged data as defined in (2); the navigation path $\mathbf{p}_i^k \subset \mathcal{A}$ is the waypoints that robot i has visited between the events; and $k > 0$ records the number. For brevity, denote by $\{\Gamma_i\} \models Q_{0,1,2}$ if the behaviors of the fleet fulfill the requirements of these requests collaboratively.

D. Problem Statement

The overall objective is to design the collaborative exploration and communication strategy, such that the total time for the operator to obtain the complete map is minimized. Additionally, the online requests from the operator should be accommodated. Thus, it is formalized as a constrained optimization over the jointed plan of all robots, i.e.,

$$\begin{aligned} \min_{\{\Gamma_i\},\, \overline{T}} \;\; & \overline{T} \\ \textbf{s.t.} \;\; & \mathcal{A} \subseteq M_{\text{h}}(\overline{T}); && \text{(5a)} \\ & (1) - (2),\;\; \forall \Gamma_i; && \text{(5b)} \\ & \{\Gamma_i\} \models Q_\ell,\;\; \forall Q_\ell \in \mathcal{Q}; && \text{(5c)} \end{aligned}$$

where $\overline{T} > 0$ is the time when the complete map $\mathcal{A}$ is known to the operator in (5a); (5b) restricts the behaviors of the fleet as described; and (5c) enforces that the set of all requests $\mathcal{Q}$ from the operator are fulfilled. Note that (5a) is overall goal of exploration which is independent of the underlying system,

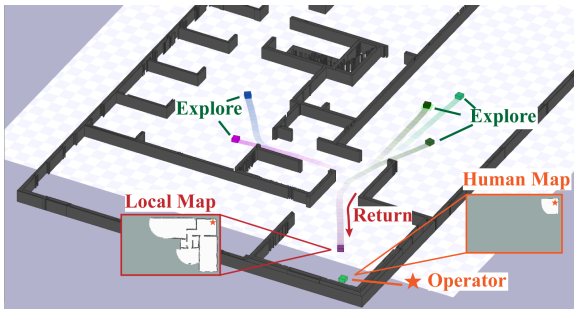

Fig. 4: Illustration of the intermittent communication protocol, including pair-wise communication and return events.

while (5b) is system-dependent. For fully-connected system without communication constraints, the constraint (5a) is often replaced by $\mathcal{A} \subseteq \bigcup_i M_i(\overline{T})$ as the union of all local maps.

Remark 2. The latency T_{h} in Q_0 could be modified online by the operator according to the exploration progress, e.g., it can be increased gradually as the fleets move further away from the operator and the newly-explored area decreases. For simplicity, the subsequent analyses are conducted for a static T_{h} first, which is then extended in Sec. III-D6. ■

III. Proposed Solution

The solution contains three main components: (i) the intermittent communication protocol as the fundamental building block; (ii) the strategy for simultaneous exploration and intermittent communication to fulfill the Q_0 and Q_1 requests; and (iii) the interactive protocol to fulfill the Q_2 request, where the operator is offered intermediate waypoints before the goal.

A. Intermittent Communication Protocol

To ensure that the information can be propagated among the team and to the operator, the robots are required to meet and communicate via the protocol of intermittent communication during exploration Guo and Zavlanos [9], Kantaros et al. [13]. To begin with, a communication topology is defined as an undirected graph $\mathcal{G} \triangleq (\mathcal{N}, \mathcal{E})$, where $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$. It satisfies two conditions: (i) $\mathcal{G}$ is connected; (ii) two robots i, j can only communicate at time t if $(i, j) \in \mathcal{E}$ and $\texttt{Qual}(\cdot)$ in (2) holds. Given a topology $\mathcal{G}$, the protocol is as follows: initially, all robots are in close proximity but each robot only communicates with its neighbors in $\mathcal{G}$; during each communication, two robots determine locally the "time and place" for their *next* communication event, and whether one robot needs to return to the operator before they meet (called a *return event*); then they depart and do not communicate until the next communication event. This procedure repeats indefinitely until termination, as shown in Fig. 4.

Definition 1 (Communication Rounds)**.** Under this protocol, all communication events can be sorted in time as rounds:

$$\mathcal{C} \triangleq C_1 C_2 \cdots C_r \cdots, \tag{6}$$

where $C_r \triangleq c_{n_1 n_1'}\, c_{n_2 n_2'} \cdots c_{n_{K_r} n_{K_r}'}$ is the r-th round of communication for $r \geqslant 1$; $c_{n_k n_k'} \triangleq (t,\, p,\, D)$ is the communication event between robots $(n_k, n_k') \in \mathcal{E}$, representing the time, location and the exchanged data, respectively; and $c_{n_k n_k'} \in C_r$, $\forall (n_k,\, n_k') \in \mathcal{E}$ and $\forall r \geqslant 1$. ■

The last condition states that all neighbors in $\mathcal{E}$ should have communicated *at least* once for each round $C_r \in \mathcal{C}$. During each round of communication, one or several return events are determined online, yielding the communication events between the robots and the operator. Similarly, let $C_{\mathrm{h}} \triangleq c^{\mathrm{h}}_{i_1}\, c^{\mathrm{h}}_{i_2} \cdots c^{\mathrm{h}}_{i_k} \cdots$ denote all the return events sorted in time, where $c^{\mathrm{h}}_{i_k} \triangleq (t_k,\, D_{i_k},\, T^{\mathrm{h}}_k,\, Q_k)$ is the communication event between robot i_k and the operator; t_k represents the time of the k-th return event; D_{i_k} denotes the data propagated to operator from robot i_k; $T^{\mathrm{h}}_k \triangleq t^{\mathrm{h}}_{k,1} t^{\mathrm{h}}_{k,2} \cdots t^{\mathrm{h}}_{k,N}$ is the time stamps of the data from each robot contained in D_{i_k}; and Q_k is the request from the operator. Each time a robot returns, the operator receives data, updates its own knowledge accordingly, and then sends requests. Given these return events, the data stored by the operator at time t is given by $D_{\mathrm{h}}(t) \triangleq \bigcup_{t_k \leqslant t} D_{i_k}$. Denote by $T_{\mathrm{h}}(t) \triangleq t^{\mathrm{h}}_1(t) t^{\mathrm{h}}_2(t) \cdots t^{\mathrm{h}}_N(t)$ the corresponding timestamps of $D_{\mathrm{h}}(t)$. It can be shown that if the condition:

$$t - t^{\mathrm{h}}_n(t) \leqslant T_{\mathrm{h}}, \tag{7}$$

holds, $\forall t \geqslant 0, \forall n \in \mathcal{N}$, then the constraint (3) is fulfilled. Moreover, the constraint (7) can be fulfilled by putting constraints on each return event, hence enabling distributed coordination for a global constraint, as stated below.

Lemma 1. *The constraint* (7) *is fulfilled, if both (i)* $t_{k+1} \leqslant T_h + \chi^h_k$*; and (ii)* $\chi^h_k < \chi^h_{k+1}$*, hold for each* $k \geqslant 0$*, where* $\chi^h_k \triangleq \mathbf{min}_n \left\{T^h_k[n]\right\}$.

Proof: See Appendix in Sec. V. ■

The first condition above indicates that each return event should occur before the largest latency exceeds the given bound, estimated by the minimal timestamp of the previous return events plus T_{h}; the second condition above ensures that the minimal stamp of each return event is strictly increasing, which in turn guarantees a monotonic decrease of the largest latency within the fleet. These two conditions combined ensure that the latency of each robot is bounded by T_{h} at all time.

B. Simultaneous Exploration and Intermit. Communication

Following the above protocol, this section describes how the communication events in (6) above are optimized online in a distributed way for the Q_0 and Q_1 requests.

1) Distributed Frontier-based Exploration: The frontier-based method in Yamauchi [26] allows a robot to explore the environment by repetitively reaching the frontiers on the boundaries between the explored and unexplored areas within its local map. Briefly speaking, given the local map $M_i(t)$ of robot $i \in \mathcal{N}$ at time $t > 0$, these boundaries can be identified via a Breadth-First-Search (BFS), which are then clustered to a few frontiers by various metrics from Holz et al. [10]. Denote by $\mathcal{F}_i(t) = \{f_k\}$ the set of frontiers, where each frontier $f_k \in \partial M_i$ is on the boundary of explored areas ∂M_i. Thus, $\mathcal{F}_i(t)$ is empty if $M_i(t)$ is fully explored.

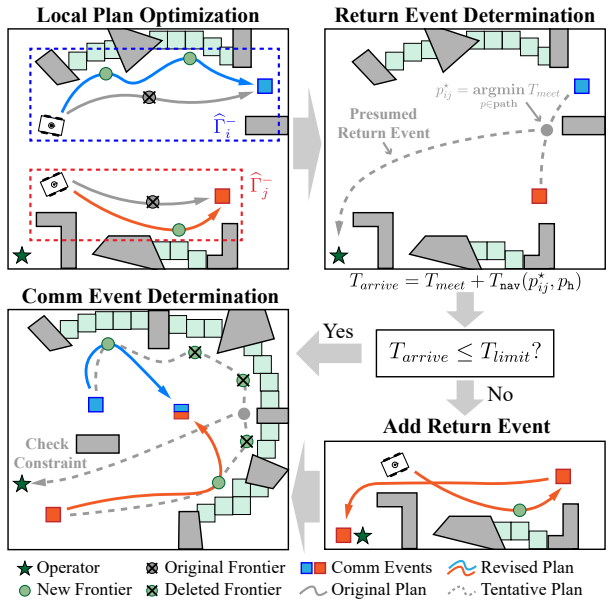

Fig. 5: Illustration of the pairwise coordination strategy during intermittent communication event.

Following the protocol of intermittent communication, consider that robots i and j communicate during the the event $c_{ij} \in C_r$ at time t. With a slight abuse of notation, the current local *plan* of robot i is given by:

$$\Gamma_i(t) \triangleq c_{ij}\mathbf{p}_i^0 c_i^1 \mathbf{p}_i^1 c_i^2 \cdots \mathbf{p}_i^{K_i} c_i^{K_i}, \tag{8}$$

where c_{ij} is the current event; c_i^k is the planned sequence of future events; $\mathbf{p}_i^k$ is the planned path between these events including visiting frontiers; and $K_i = |\mathcal{N}_i|$ is the number of neighbors for robot i in $\mathcal{G}$. Note that the time and place for the events $\{c_i^k\} \subset \Gamma_i(t)$ are already confirmed, thus *should not* be altered. However, the intermediate paths $\{\mathbf{p}_i^k\}$ can be modified by choosing a different path to visit more frontiers, as shown in Fig. 5. More specifically, denote by $\widehat{\Gamma}_i(t) \triangleq \widehat{\mathbf{p}}_i^0 c_i^1 \widehat{\mathbf{p}}_i^1 c_i^2 \cdots \widehat{\mathbf{p}}_i^{K_i} c_i^{K_i} \widehat{\mathbf{p}}_{ij} \widehat{c}_{ij}$ the *revised* plan of robot i, where $\{\widehat{\mathbf{p}}_i^k\}$ are the updated paths and $\widehat{\mathbf{p}}_{ij}$ is the newly-added path before the next event $\widehat{c}_{ij}$. Thus, the sub-problem of local coordination can be formulated as follows.

Problem 1. Given the local data (Γ_i, M_i) and (Γ_j, M_j), the objective is to synthesize the revised plans $\widehat{\Gamma}_i, \widehat{\Gamma}_j$ such that: (i) the total number of frontiers that are planned to be visited by robots i, j is maximized; and (ii) the largest latency δ_r in (7) for the current round is bounded by T_{h}. ■

Note that the first objective is defined locally for robots i, j, whereas the second objective is a global constraint. The remaining part describes how it can be solved in a distributed way via intermittent communication and local optimization.

2) Optimization of Local Plans: To begin with, robot i executes event c_{ij} in (8) by meeting with robot j around time t_{ij} and at location p_{ij}. The exchanged data from robot i is given by:

$$D_i(t) \triangleq (M_i, \Gamma_i, \mathcal{F}_i', \Omega_i, H_i), \tag{9}$$

where M_i, Γ_i are defined earlier; $\mathcal{F}_i' \triangleq \mathcal{F}_i \cup \mathcal{F}_i^-$ is the set of frontiers known to robot i, with $\mathcal{F}_i$ being the frontiers assigned to robot i and $\mathcal{F}_i^-$ the frontiers known to robot i but assigned to *other* robots; $\Omega_i \triangleq (\Omega_i^{\mathrm{l}}, \Omega_i^{\mathrm{h}})$ is a tuple of two vectors with length N, i.e., $\Omega_i^{\mathrm{l}} \triangleq \omega_1^{\mathrm{l}}\omega_2^{\mathrm{l}}\cdots\omega_N^{\mathrm{l}}$ is the estimated timestamps of the data from robot $n \in \mathcal{N}$ that is known to robot i after its last communication event $c_i^{K_i}$; and $\Omega_i^{\mathrm{h}} \triangleq \omega_1^{\mathrm{h}}\omega_2^{\mathrm{h}}\cdots\omega_N^{\mathrm{h}}$ is the estimated timestamps of data from robot $n \in \mathcal{N}$ that is known to the operator at time $t_i^{K_i}$; $H_i \triangleq \{(\mathcal{P}_q, p_{\mathrm{h}}^\star, t_{\mathrm{h}})\}$ contains the requests from the operator, including the prioritized region $\mathcal{P}_q \subseteq Q_1$, the desired position $p_{\mathrm{h}}^\star \in Q_2$, and t_{h} is the timestamp for this request. Note that Ω_i^{l} and Ω_i^{h} are initialized as zeros and updated each time when a communication event is confirmed. The local data D_j of robot j is defined analogously.

After D_i and D_j are exchanged, these data is processed by robot i as follows. First, the merged map of M_i and M_j is computed as its new local map, i.e., $M_{ij} \triangleq \texttt{merge}(M_i, M_j)$. Then, the frontiers $\mathcal{F}_{ij}$ associated with the merged M_{ij} is computed, and the complete set of frontiers that can be visited is given by $\widehat{\mathcal{F}} \triangleq \mathcal{F}_{ij} \backslash \big((\mathcal{F}_i^- \backslash \mathcal{F}_j) \cup (\mathcal{F}_j^- \backslash \mathcal{F}_i)\big)$, which excludes the frontiers in $\mathcal{F}_{ij}$ that have been assigned to other robots in the previous events. Next, a new vector of timestamps Ω_{ij}^{h} is calculated by taking the maximum of Ω_i^{h} and Ω_j^{h} to derive the estimated stamps of each robot known to operator, i.e.,

$$\Omega_{ij}^{\mathrm{h}}[n] \triangleq \mathbf{max}\Big\{\Omega_i^{\mathrm{h}}[n],\, \Omega_j^{\mathrm{h}}[n]\Big\}, \tag{10}$$

where $\Omega_i^{\mathrm{h}}[n] = \omega_n^h$ is the timestamp of the data from robot $n \in \mathcal{N}$ as defined in (9). It serves as an estimation of the timestamps $T_{\mathrm{h}}(t)$ for the operator in this round, which is essential to determine whether a return event is needed later. Lastly, the requests H_i are merged with H_j by the timestamps.

Remark 3. The local map merging described above can be done efficiently based on [1], often with a common coordinate system that is agreed-upon beforehand, or via map fusion and alignment during execution as proposed in [15, 24]. Tackling uncertainty and misalignment when merging local maps with varying accuracy is outside the scope of this work. ■

Given $\widehat{\mathcal{F}}$ above, a planning algorithm consists of three main steps: (i) the frontiers in $\widehat{\mathcal{F}}$ are divided into two clusters $\widehat{\mathcal{F}}_i, \widehat{\mathcal{F}}_j$ associated with either robot i or j, e.g., by comparing the minimum distance from $f \in \widehat{\mathcal{F}}$ to the confirmed communication locations in Γ_i and Γ_j; (ii) a traveling salesman problem with time windows (TSPTW) is formulated for each robot locally to optimize the sequence of frontiers to visit between communication events, i.e., the paths $\{\widehat{\mathbf{p}}_i^k\}$ for robot i to visit $\widehat{\mathcal{F}}_i$ (the same for robot j). Detailed modeling is omitted here due to limited space. The resulting preliminary plan is given by $\widehat{\Gamma}_i' = \widehat{\mathbf{p}}_i^0 c_i^1 \widehat{\mathbf{p}}_i^1 c_i^2 \cdots \widehat{\mathbf{p}}_i^{K_i} c_i^{K_i} \mathbf{p}_i^{K_i+1}$, which is *not* yet the final revised plan $\widehat{\Gamma}_i$ as the communication event $\widehat{c}_{ij}$ has not been determined; (iii) the next communication event $\widehat{c}_{ij}$ is optimized to fulfill (7), given $\widehat{\Gamma}_i'$ and $\widehat{\Gamma}_j'$.

Remark 4. The first two steps above can *not* be replaced by formulating and solving a multiple-vehicle routing problem

with time window (MVRPTW). This is due to the fact that the communication events are *already assigned* to respective robots and only the frontiers between these events can be shuffled subject to the time windows. ■

3) Next Communication Event: To ensure that the latency $\delta_r \leqslant T_{\mathrm{h}}$, an additional constraint regarding the next communication event $\widehat{c}_{ij}$ is added, i.e.,

$$\widehat{t}_{ij} + T_i^{\mathrm{nav}}(\widehat{p}_{ij}, p_{\mathrm{h}}) \leqslant T_{\mathrm{h}} + \mathbf{min}_n \left\{\Omega_{ij}^{\mathrm{h}}[n]\right\}, \tag{11}$$

where $\widehat{t}_{ij}$, $\widehat{p}_{ij}$ are the expected meeting time and location, respectively; p_{h} is the current location of the operator; $T_i^{\mathrm{nav}}(p_1, p_2)$ is the estimated duration for robot i to travel from location p_1 to p_2 within its map M_i, (which is set to ∞ if p_2 is un-reachable); Ω_{ij}^{h} is the vector of timestamps as defined in (10). It indicates that if robot i meets the operator directly after meeting robot j, the operator can receive the data from each robot $n \in \mathcal{N}$ with a latency less than T_{h}. Note that Ω_{ij}^{h} in constraint (11) is updated according to the last confirmed communication event in $\widehat{\Gamma}_i'$ and $\widehat{\Gamma}_j'$. Moreover, since the meeting events in the preliminary plan $\widehat{\Gamma}_i'$ are already fixed, it can be divided into two segments: $\widehat{\Gamma}_i^- \triangleq \widehat{\mathbf{p}}_i^0 c_i^1 \cdots \widehat{\mathbf{p}}_i^{K_i} c_i^{K_i}$ and $\mathbf{p}_i^{K_i+1}$(the same for robot j). In the final plan, the segment $\widehat{\Gamma}_i^-$ is reserved, while the remaining frontiers, denoted by $\widehat{\mathcal{F}}_{ij}$, are re-assigned as intermediate waypoints between the events $c_i^{K_i}$ and $\widehat{c}_{ij}$ for robot i (between $c_j^{K_j}$ and $\widehat{c}_{ij}$ for robot j). Thus, the final plan is given by $\widehat{\Gamma}_i \triangleq \widehat{\Gamma}_i^- + \widehat{\Gamma}_i^+$, where $\widehat{\Gamma}_i^+ \triangleq \widehat{\mathbf{p}}_i^{K_i+1}\widehat{c}_{ij}$ and $\widehat{\Gamma}_j^+ \triangleq \widehat{\mathbf{p}}_j^{K_j+1}\widehat{c}_{ij}$ are the tailing segments to be optimized as stated below.

Problem 2. Given $\widehat{\Gamma}_i^-$, $\widehat{\Gamma}_j^-$ and $\widehat{\mathcal{F}}_{ij}$, $\widehat{\Gamma}_i^+$, $\widehat{\Gamma}_j^+$ should be chosen such that: (i) the number of frontiers contained in $\widehat{\Gamma}_i^+, \widehat{\Gamma}_j^+$ is maximized; and (ii) condition (11) holds for $\widehat{c}_{ij}$. ■

The proposed solution is summarized in Alg. 1. It consists of two main parts: (i) to determine whether robot i (or robot j) should return to the operator as a return event, *after* its latest confirmed communication event $c_i^{K_i}$ (or $c_j^{K_j}$) and *before* the communication event $\widehat{c}_{ij}$; (ii) to optimize the next communication event $\widehat{c}_{ij}$ given the return event.

Return event. Since both $c_i^{K_i}$ and $c_j^{K_j}$ belong to the merged map M_i, an obstacle-free shortest path $\overline{\mathbf{p}}_{ij}$ can be found by a path planning algorithm from $c_i^{K_i}$ to $c_j^{K_j}$ (Line 2). Then, a point $p_{ij}^\star$ within $\overline{\mathbf{p}}_{ij}$ is determined by minimizing the meeting time if $p_{ij}^\star$ is chosen as the meeting location, i.e.,

$$p_{ij}^\star = \underset{p \in \overline{\mathbf{p}}_{ij}}{\mathbf{argmin}} \left\{ \underset{\ell=i,j}{\mathbf{max}} \left\{ t_\ell^{K_\ell} + T_\ell^{\mathrm{nav}}(p_\ell^{K_\ell}, p) \right\} \right\}, \tag{12}$$

of which the associated time is denoted by $t_{ij}^\star$ (Line 3); and $T_\ell^{\mathrm{nav}}(\cdot)$ is defined in (11). Then, if either robot i or j departs from $p_{ij}^\star$ at time $t_{ij}^\star$ and immediately returns to the operator, the latency constraint can be checked by (11). If it does not hold for either robot i or j, it indicates that either one needs to return to the operator before they meet (Line 4). In this case, the robot that minimizes the largest latency is selected to return; and if both robots can't reduce largest latency, both will have to return. For now, assumed that robot j returns to the operator after its latest event $c_j^{K_j}$. Since this return event is a meeting event at p_{h}, it can be appended to $\widehat{\Gamma}_j^-$ as an additional event (Line 5), i.e., the new $C_j^{K_j}$ now denotes the return event. Consequently, Ω_{ij}^{h} is updated by:

$$\Omega_{ij}^{\mathrm{h}}[n] = \begin{cases} \mathbf{max}\left\{\Omega_{ij}^{\mathrm{h}}[n], \Omega_j^{\mathrm{l}}[n]\right\}, & n \neq j; \\ t_j^{K_j} + T_i^{\mathrm{nav}}(p_j^{K_j}, p_{\mathrm{h}}), & n = j, \end{cases} \tag{13}$$

which means that the timestamps of the data that the operator possesses will be updated when robot j returns (Line 6-7). On the other hand, if neither robot i or j should return before they meet, their communication event is optimized in the following.

Communication event. To begin with, a traveling salesman problem (TSP) is formulated by setting $p_i^{K_i}$ as the starting depot, $p_j^{K_j}$ as the ending location, and frontiers in $\widehat{\mathcal{F}}_{ij}$ as the waypoints to visit (Line 10). Its solution is denoted by $\widehat{\mathbf{F}}_{ij}$ as a sequence of frontiers. Then, the next communication event is optimized by an iterative algorithm as follows. In each iteration, a collision-free shortest path $\widehat{\mathbf{p}}_{ij}$ is generated that starts from $p_i^{K_i}$, traverses the frontiers in $\widehat{\mathbf{F}}_{ij}$ in order, and ends at $p_j^{K_j}$ (Line 11). Within $\widehat{\mathbf{p}}_{ij}$, a potential meeting point $\widehat{p}_{ij}$ and time $\widehat{t}_{ij}$ is determined by (12) (Line 12). Consider the following two cases: (i) If the result satisfies the constraint (11), the meeting event $\widehat{c}_{ij}$ is set to $(\widehat{p}_{ij}, \widehat{t}_{ij})$ and the iteration stops; (ii) However, if none of the waypoints in $\widehat{\mathbf{p}}_{ij}$ can satisfy the constraint (11) (Line 9), the frontier in $\widehat{\mathcal{F}}_{ij}$ with the maximum cost is removed from $\widehat{\mathcal{F}}_{ij}$ (Line 13-14). Then,

Algorithm 1: Optimize Next Communication Event

Input : $\widehat{\Gamma}_i^-$, $\widehat{\Gamma}_j^-$, $\widehat{\mathcal{F}}_{ij}$.
Output: $\widehat{\Gamma}_i^+$, $\widehat{\Gamma}_j^+$.
/* **Return event** */
1 $T_{\mathtt{lim}} \leftarrow T_{\mathrm{h}} + \mathbf{min}_n \left\{\Omega_{ij}^{\mathrm{h}}[n]\right\}$;
2 $\overline{\mathbf{p}}_{ij} \leftarrow \mathtt{GenPath}(p_i^{K_i}, p_j^{K_j})$;
3 $c_{ij}^\star \leftarrow \mathtt{SelComm}(c_i^{K_i}, c_j^{K_j}, \overline{\mathbf{p}}_{ij})$;
4 **if** *not* $\mathtt{CheckConst}(c_{ij}^\star, T_{lim})$ **then**
5 $\quad c_j^{K_j} \leftarrow \mathtt{CommEvent}(p_{\mathrm{h}}, t_j^{K_j} + T_i^{\mathrm{nav}}(p_j^{K_j}, p_{\mathrm{h}}))$;
6 $\quad \Omega_{ij}^{\mathrm{h}} \leftarrow \mathtt{UpdTimeStp}(\Omega_{ij}^{\mathrm{h}}, \Omega_j^{\mathrm{l}})$;
7 $\quad T_{\mathtt{lim}} \leftarrow T_{\mathrm{h}} + \mathbf{min}_n \left\{\Omega_{ij}^{\mathrm{h}}[n]\right\}$;
/* **Next comm. event** */
8 $\widetilde{\mathcal{F}}_{ij} = \widehat{\mathcal{F}}_{ij}$;
9 **while** *not* $\mathtt{CheckConst}(\widehat{c}_{ij}, T_{lim})$ **do**
10 $\quad \widehat{\mathbf{F}}_{ij} \leftarrow \mathtt{TSP}(p_i^{K_i}, \widetilde{\mathcal{F}}_{ij}, p_j^{K_j})$;
11 $\quad \widehat{\mathbf{p}}_{ij} \leftarrow \mathtt{GenPath}(p_i^{K_i}, \widehat{\mathbf{F}}_{ij}, p_j^{K_j})$;
12 $\quad \widehat{c}_{ij} \leftarrow \mathtt{SelComm}(c_i^{K_i}, c_j^{K_j}, \widehat{\mathbf{p}}_{ij})$;
13 $\quad f^\star \leftarrow \mathbf{argmax}_{f \in \widetilde{\mathcal{F}}_{ij}} \{cost(f)\}$;
14 $\quad \widetilde{\mathcal{F}}_{ij} \leftarrow \widetilde{\mathcal{F}}_{ij} \backslash \{f^\star\}$;
15 $\widehat{\mathbf{p}}_i^{K_i+1}, \widehat{\mathbf{p}}_j^{K_j+1} \leftarrow \mathtt{Split}(\widehat{\mathbf{p}}_{ij}, \widehat{c}_{ij})$;
16 $\widehat{\Gamma}_i^+ \leftarrow \widehat{\mathbf{p}}_i^{K_i+1} + \widehat{c}_{ij}$;
17 $\widehat{\Gamma}_j^+ \leftarrow \widehat{\mathbf{p}}_j^{K_j+1} + \widehat{c}_{ij}$;

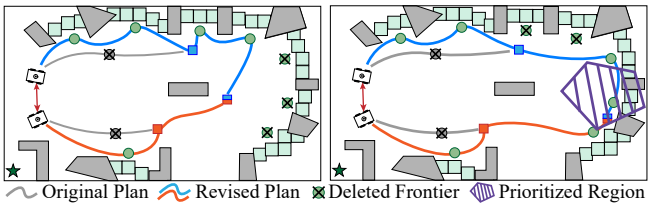

Fig. 6: Illustration of the revised plans with (left) and without (right) the prioritized region.

the iteration continues. Given a frontier $f \in \widehat{\mathcal{F}}_{ij}$, its cost is calculated in two cases: First, if there is no prioritized region given by the operator, it is defined by:

$$cost(f) \triangleq T_i^{\text{nav}}(f, p_{\text{h}}) + \mathbf{max}_{n\in\{i,j\}}\{T_n^{\text{nav}}(f, p_n^{K_n}))\}, \quad (14)$$

as the estimated time for robots i or j to reach f and then return to the operator; Second, if there is a prioritized region by Q_1 request, the cost is given by $cost(f) \triangleq T_i^{\text{nav}}(f, p_c)$, where p_c is the center of the prioritized region, as shown in Fig. 6. As proven in Lemma 2, the above procedure terminates in finite steps.

Thus, the respective paths $\mathbf{p}_i^{K_i+1}$ and $\mathbf{p}_j^{K_j+1}$ before $\widehat{c}_{ij}$ in $\widehat{\Gamma}_i^+$ and $\widehat{\Gamma}_j^+$ are determined by splitting $\widehat{\mathbf{p}}_{ij}$ into two segments at $\widehat{p}_{ij}$ (Line 15-17). Lastly, once the final plans $\widehat{\Gamma}_i$ and $\widehat{\Gamma}_j$ are determined, data D_i in (9) is updated with the revised Γ_i and $\mathcal{F}_i$ (the same for robot j). More importantly, the timestamps in Ω_i are updated given this new meeting event, i.e., $\Omega_i^{\text{h}}[n] = \Omega_{ij}^{\text{h}}[n]$, where $\Omega_{ij}^{\text{h}}[n]$ is updated in (13); and

$$\Omega_i^{\text{l}}[n] = \begin{cases} \widehat{t}_{ij},\ n \in \{i,j\}; \\ \mathbf{max}\left\{\Omega_i^{\text{l}}[n], \Omega_j^{\text{l}}[n]\right\},\ n \neq i,j; \end{cases} \quad (15)$$

where $\widehat{t}_{ij}$ is the confirmed meeting time. Note that Ω_j is updated in the same way by robot j.

Remark 5. The derived $\widehat{\Gamma}_i^+$ and $\widehat{\Gamma}_j^+$ are optimal given the information available to robots i, j in this round, i.e., the number of frontiers in $\widehat{\Gamma}_i^+$ and $\widehat{\Gamma}_j^+$ is maximized subject to the constraints on the next communication event in (11). ■

C. Dynamic Movement of Operator

The Q_2 request in (5) specifies that the operator wants to move to another location within the explore area. Indeed, if the operator remains static, due to the latency constraint by T_{h} and the scarce communication, the area that a robotic fleet can explore by *any strategy* is strictly bounded. i.e., the boundary of explored area to the operator is upper-bounded by:

$$d_{\max} \leqslant (1 - \frac{1}{2^N})T_{\text{h}} v_{\max} + N d_{\text{com}}, \quad (16)$$

where $v_{\max}$ is the maximum speed of all robots; N is the total number of robots; and d_{comm} is the maximum range of communication. An illustration of how the explored area changes w.r.t. the number of robots is shown in Fig. 7. It can be seen that the exploration efficiency drastically decreases as the robots are close to above boundary. Consequently, if the operator remains static, it often happens that the deployed fleet can not explore the workspace fully.

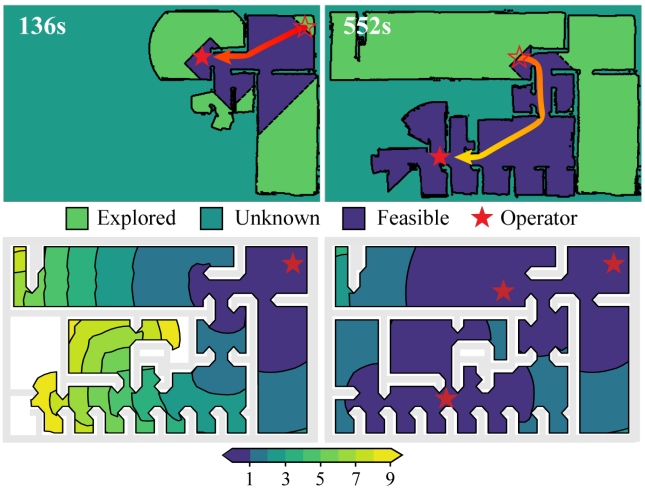

Fig. 7: Top: Illustration of the feasible region for the operator defined in (17) at different time instants. **Bottom**: Comparison of the maximum explored area when the operator is static (**Left**) or dynamic (**Right**), where the color bar indicates the number of robots required.

Therefore, the operator is allowed to move within the explored area. However, the operator can not move arbitrarily, due to the same latency constraints in (7). More specifically, each time robot $i^\star \in \mathcal{N}$ returns, the operator can choose a new pose p'_{h} within the latest map, under the following constraint:

$$T_i^{\text{nav}}(p'_{\text{h}}, p_i^{k_i}) \leqslant T_i^{\text{nav}}(p_{\text{h}}, p_i^{k_i}),\ \forall k_i \leqslant K_i,\ \forall i \in \mathcal{N}; \quad (17)$$

where p_{h} is the current pose of the operator; K_i is the number of meeting events for robot i; $p_i^{k_i}$ is the pose of the k_i-th communication event for robot i out of its K_i events. It requires that the new pose p'_{h} should be no further to the confirmed meeting events (in terms of navigation time for robot i) than the current pose p_{h}. In this way, the condition (11) is still fulfilled if the operator moves to p'_{h}.

Definition 2 (Feasible Region)**.** The set of all new poses of the operator p'_{h} that satisfy (17) is called the *feasible region* and denoted by $\mathcal{P}_{\text{h}}$. ■

In other words, if the operator sets $p_{\text{h}}^\star$ as the Q_2 request that is outside of the feasible region, the path to $p_{\text{h}}^\star$ is decomposed into several segments via intermediate waypoints such that the subsequent waypoint is always feasible once the previous one is reached. The operator can choose among these waypoints and move gradually the desired $p_{\text{h}}^\star$. For instance, once the new pose p'_{h} is chosen, it is sent to any return robot, and then the operator moves to p'_{h}. This updated operator pose is broadcast to other robots via the subsequent communication events among the robots, such that the robots can plan the next return events based on the updated pose. This process is repeated until the operator reaches $p_{\text{h}}^\star$, as shown in Fig. 7.

Remark 6. Note that the operator behavior of choosing the next new pose can be sub-optimal (different from the

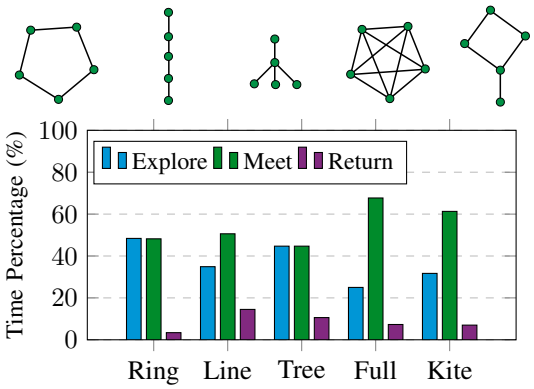

Fig. 8: Comparison of exploration efficiency (**Bottom**) under different communication topologies (**Top**), for the small office in Sec. IV with 5 robots and $T_{\mathrm{h}} = 150s$.

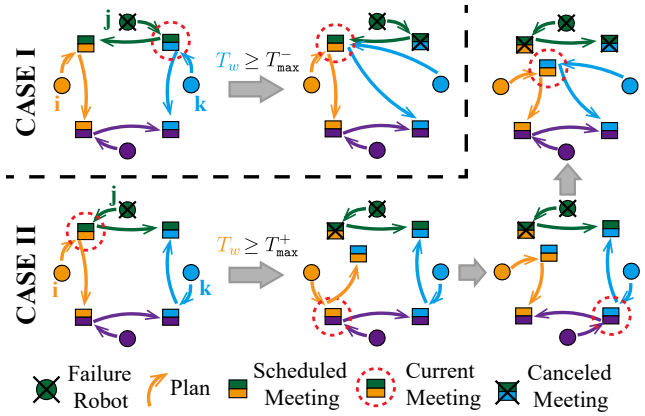

Fig. 9: Illustration of two cases during the failure detection and recovery mechanism in Sec. III-D5.

planned p'_{h}) or even arbitrary, as long as it belongs to the feasible region $\mathcal{P}_{\mathrm{h}}$ from Def. 2. ■

D. Discussion

1) Correctness: The existence of a solution by Alg. 1 and the guarantee on the Q_0 request are stated below, of which the proofs are given in the Appendix of Sec. V.

Lemma 2. *Alg. 1 always returns a feasible solution during each intermittent communication for all rounds.*

Theorem 1. *Under the proposed overall strategy, the latency constraint by T_h in* (3) *is fulfilled.*

2) Choice of Communication Topology: Although the proposed strategy above is valid for any connected communication topology $\mathcal{G}$, a fixed ring topology among the robots is adopted to ensure a high efficiency of information propagation, while a dynamic connection between the robots and the operator is allowed. More specifically, the resulting schedule of communication C_0 in (6) satisfies that $n'_i = n_{i+1}$, $\forall n_i \in C_0$, i.e., robot n'_i is simply the succeeding robot of n_i, and n_i is the preceding robot of n'_i. Whenever a return event is required by (12) during the meeting event of two robots, the preceding robot always returns to the base. It can be shown that under this topology, the data from any robot can be transmitted to any other robot after maximum N meeting events. More importantly, this topology ensures that when two robots meet, they would have obtained the information from *all* previous meeting events. This feature yields the ring topology particularly suitable for the scenarios of human-robot interaction, where the operator can be informed about the overall progress of the whole fleet when a single robot returns. Second, this topology is essential for the Q_2 request, due to the following two aspects: (I) The calculation of feasible region $\mathcal{P}_{\mathrm{h}}$ in Def. 2 relies on the knowledge of meeting events of all robots, which can be easily obtained under the ring topology; (II) After a Q_2 request is sent and the operator moves away, the ring topology ensures that any other robot that is scheduled to return can be informed about the new operator position, before it actually returns. Lastly, as shown in Fig. 8 and later in Sec. IV, the ring topology often leads to a higher rate of time spent on exploration and lower rate on returning to the operator, compared with other topologies.

3) Local Plan Adaptation: During the online execution, it is possible that a frontier that is assigned to one robot is explored earlier by another robot on its way to other frontiers. In addition, it might have generated additional frontiers and there is sufficient time left before the next meeting event. More importantly, the preliminary plans Γ_i, Γ_j derived via Alg. 1 may exceed the planned meeting time due to motion uncertainties, which may lead to excessive waiting time and a potential violation of the Q_0 request. Therefore, it is important for robots to update their local plans as follows. Each time when robot $i \in \mathcal{N}$ reaches a frontier point, it extracts the set of frontiers $\mathcal{F}_t$ within its latest map, among which the condition $T_i^{\mathrm{nav}}(p_i, f) + T_i^{\mathrm{nav}}(f, p_c) < t_c - t$ is checked for each $f \in \mathcal{F}_t$, with p_i being the current robot pose, p_c and t_c the pose and time of the next communication event. If so, it ensures that robot i can reach p_c before t_c if it visits the frontier f. Then, the priority of remaining frontiers is measured by $\chi(f) \triangleq \omega_1 \mathbf{mean}_{f^- \in \mathcal{F}_i^-}\{T_i^{\mathrm{nav}}(f, f^-)\} - \omega_2 T_i^{\mathrm{nav}}(p, f) - \omega_3 \mathbf{mean}_{f^+ \in \mathcal{F}_i}\{T_i^{\mathrm{nav}}(f, f^+)\}$, where $\mathcal{F}_i^-$ are frontiers allocated to other robots; $\mathcal{F}_i$ are frontiers allocated to robot i; and $\omega_1, \omega_2, \omega_3 \geqslant 0$ are parameters. Thus, frontiers are prioritized if they are close to the current robot pose and the frontiers already assigned to robot i, and far from frontiers assigned to other robots. The frontier with the highest priority is chosen as the next goal point. If no frontiers satisfy this condition, robot i moves to the next meeting event directly.

4) Spontaneous Meeting Events: As the navigation path is determined by the SLAM module in (1), it often occurs that robot i meets with another robot k on its way to the meeting event with robot j, which is called a *spontaneous* meeting event. In this case, they exchange their local data D_i and D_k including merging the local maps. However, they do not coordinate the next meeting event nor modify their local plans, such that the communication topology $\mathcal{G}$ can be maintained, instead of growing into a full graph.

5) Failure Detection and Recovery: Robots can often fail when operating in extreme environments as considered in this work. Thus, a failure detection and recovery mechanism is

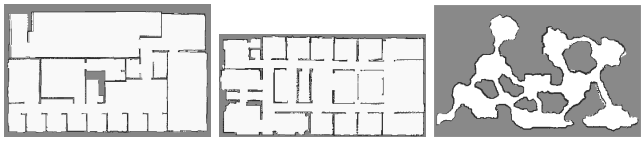

Fig. 10: Test environments: large office with three separated long corridors (**Left**), small office with more connected rooms (**Middle**), and subterranean cave (**Right**).

essential for the safety of the functional robots. The key is to ensure that the communication topology can be dynamically adjusted to form a smaller ring after one robot fails, still via only intermittent local communication. Assume that robot j fails at time $t_f > 0$, and its proceeding and succeeding neighbors in $\mathcal{G}$ are robot i and robot k, respectively. As illustrated in Fig. 9, the mechanism can be sorted into two cases: (I) If robot j fails on its way to meet robot i, robot i will wait for robot j at the communication location c_{ji} agreed from the previous round, for a waiting period of $T^{-}_{\max}$. Then, robot i assumes that robot j has failed and *directly* moves to the meeting location c_{jk} of robots j and k, which has been propagated to robot i via previous communication events. During communication, robots i and k plans for their next meeting event as described previously; (II) If robot j fails *before* meeting with robot k, robot k detects the failure of robot j in the same way as robot i does in the first case, with a waiting time $T^{+}_{\max}$. However, robot k in this case cannot directly meet with robot i, as robot i has already communicated with robot j and most likely performs exploration. Moreover, the meeting event between robot i and its predecessor is unknown to robot k due to the failure of robot j. Therefore, robot k first determines a meeting location p_{ki} locally, and then communicates with its successor to propagate this information. After several communication events, this location is propagated to robot i along with the message that robot j has failed. Thus, robot i cancels the meeting event with robot j and heads to p_{ki} to meet with robot k instead. In both cases, the communication topology $\mathcal{G}$ is adjusted to form a smaller ring, by removing the failed robot. It is worth noting that $T^{+}_{\max}$ should be larger than $T^{-}_{\max}$, such that robot i can meet with robot k in case (I), before robot k judges that robot j has failed and moves away.

6) Time-varying Latency Constraint: During online execution, the operator may require different frequency of status updates according to the current progress or other factors. Therefore, the proposed framework allows the operator to dynamically adjust the constraint T_{h} as the Q_0 request, i.e., by specifying a new threshold T'_{h} for the future rounds when a robot returns. Thus, this new threshold is propagated to the fleet and is reflected automatically in the subsequent intermittent communication events.

IV. Numerical Experiments

To further validate the proposed method, numerical simulations and hardware experiments are presented in this section. The proposed method is implemented in `Python3` within the framework of `ROS`, and tested on a laptop with an Intel Core i7-1280P CPU. The operator interacts with the robotic fleet via an Android tablet. Simulation and experiment videos can be found in the supplementary material.

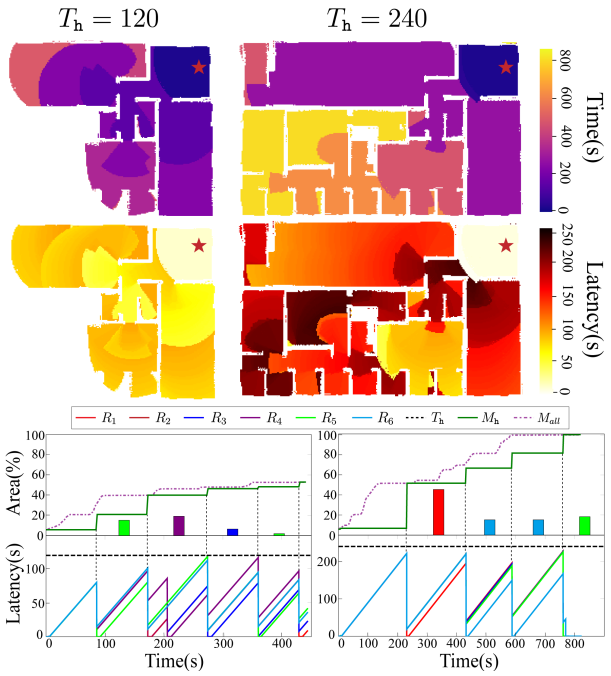

Fig. 11: Simulation results of 6 robots exploring a $50m \times 30m$ office, when the latency constraint T_{h} is set to $120s$ (**Left**) and $240s$ (**Right**). **Top**: the final map at the operator (marked by the red star), color-filled by the time of being received; **Middle**: the latency of different parts, measured by the time difference between being explored and being received by the operator; **Bottom**: evolution of the union of all explored maps $M_{\mathtt{all}}$, the local map M_{h} of operator, the contribution of return robots, and the latency of each robot in (3).

A. System Description

The robotic fleet consists of 6 differential-driven UGVs, which is simulated in the `Stage` simulator for three different environments as shown in Fig. 10: (i) a large office of size $50m \times 30m$, with three separated long corridors; (ii) a smaller but more connected office of size $40m \times 20m$; (iii) a subterranean cave of size $50m \times 30m$. A 2D occupancy grid map is generated via the `gmapping` SLAM package to generate local map, with a sensor range of $8m$, which provides a probabilistic representation of the environment by assigning values to each cell, i.e., whether the cell has been explored, free or occupied by obstacles [17]. Each robot navigates using the ROS navigation stack `move_base`, with a maximum linear velocity of $0.5m/s$ and angular velocity of $0.8rad/s$.

Moreover, two robots can only communicate with each other if they are within a communication range of $3.5m$ and have a line of sight, the same between robots and the operator. The map merge during communication is handled by the off-the-shelf ROS package `multirobot_map_merge`. Lastly, the operator interacts with the robotic fleet through the terminal

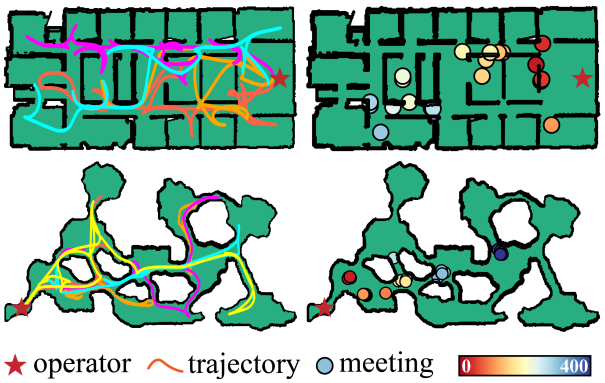

Fig. 12: Robot trajectories within the final map (**Left**)s and the communication points (**Right**) in the small office (**Top**) and the cave (**Bottom**), with $T_{\mathrm{h}} = 150s$ and $T_{\mathrm{h}} = 200s$. Meeting points are color-filled by the meeting time (red to blue).

and the `Rviz` interface. Namely, the operator can send a request by first specifying its type (Q_1 or Q_2) in the terminal; and then either select the vertices of prioritized region with the `Publish_Point` tool, or choose the next desired location with the `2D_Nav_Goal` tool.

B. Results

The results present: (i) how the proposed method performs under different environments and latency constraints; (ii) how the human-robot interactions (i.e., Q_1 and Q_2 requests) are handled during the exploration process.

1) Latency-constrained Exploration: As shown in Fig. 11, the operator is located at the top-right corner of the the first large office, which is also where all robots start. The latency constraint T_{h} is set to $120s$ and $240s$ for two cases. It can be seen that the fleet can not explore the whole environment when $T_{\mathrm{h}} = 120s$ with only 53% coverage, while the whole map is obtained when $T_{\mathrm{h}} = 240s$. Moreover, the latency constraint is satisfied at all time in both cases, with the maximum latency among all robots being $118s$ and $227s$, respectively. In other words, the explored map is updated to operator in time for both cases, and a smaller T_{h} leads to much more frequent updates. Specifically, when $T_{\mathrm{h}} = 120s$, the return events occur at $87s$, $173s$, $207s$, $275s$, $361s$ and $431s$, during which the newly-explored map is increased by 15%, 19%, 0%, 6%, 2% and 4%, respectively. Thus, the efficiency quickly decreases as the explored area extends, i.e., most of the time is spent on navigation and communication, rather than exploration. Similar phenomenon can be observed when $T_{\mathrm{h}} = 240s$, where the operator map is increased by 45%, 15%, 15%, 18% in four return events. Lastly, it is worth noting that although the whole map is obtained when $T_{\mathrm{h}} = 240s$, the operator can not obtain a timely update of the fleet status due to the large latency, i.e., only at $232s$, $432s$, $588s$ and $762s$. The left-bottom area is the latest to be explored at $592s$. This further validates that simply increasing T_{h} can not guarantee a timely update of the fleet status, thus movement of operator is essential.

In addition, the second small office is tested with 4 robots and $T_{\mathrm{h}} = 150s$, while the cave environment is tested with 5 robots and $T_{\mathrm{h}} = 200s$. The final trajectories and communication points are shown in Fig. 12. It can be seen that the whole map is explored for both scenes, at time $243s$ and $395s$, respectively. Different from the first large office, the communication points gradually move forward as the exploration proceeds, as the robots often navigate a short distance to reach the meeting points, yielding more time for exploration. It indicates that the proposed method is particularly effective for a more connected environment.

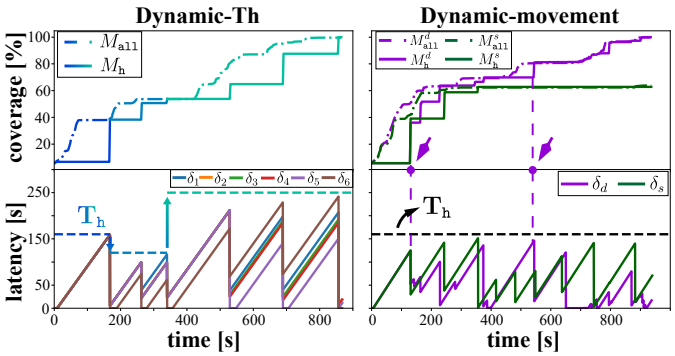

Fig. 13: Left: Evolution of the explored area $M_{\mathtt{all}}$, the local map of the operator M_{h}, and the latency of each robot, when T_{h} changes dynamically; **Right**: Comparison of the exploration efficiency when the operator is static (green) or dynamic (purple) (**Top**), and evolution of the largest latency over all robots (**Bottom**), with the arrows indicating when the operator moves.

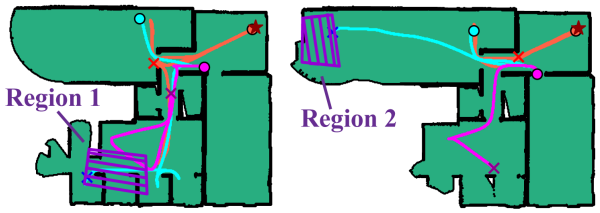

Fig. 14: Comparison of the exploration results when different operators specify different Q_1 requests as prioritized regions (the purple polygon). The trajectories (colored lines) start from the communication points (filled circles), and end when the prioritized region is fully explored (crosses).

2) Human-robot Interaction: As the operator may interact with the fleet via three different types of requests, this section evaluates how the proposed method handles these interactions during exploration.

(I) **Time-varying latency constraint in** Q_0. To begin with, regarding the Q_0 requests where the operator changes T_{h}, the large office is tested with 6 robots and an initial T_{h} of $160s$. The results are summarized in Fig. 13, which show that T_{h} is changed twice. At $t = 158s$, the operator decreases T_{h} to $120s$, yielding more frequent updates and smaller exploration efficiency; while at $t = 330s$, T_{h} is increased to $250s$, after which exploration efficiency quickly increases with less frequent return events. Lastly, the whole map is obtained at $860s$, during which the maximum latency of all robots never exceeds the bound T_{h}.

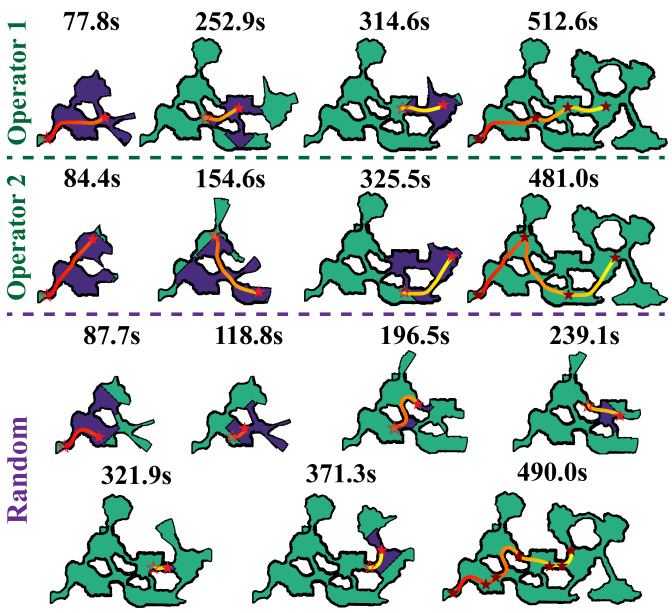

Fig. 15: Results of how the proposed method reacts to different human behaviors of dynamic movement: two operators (**Top**) and a random movement (**Bottom**). Human movements are shown as colored lines with intermediate goal positions (red stars) and feasible regions for each movement (purple area).

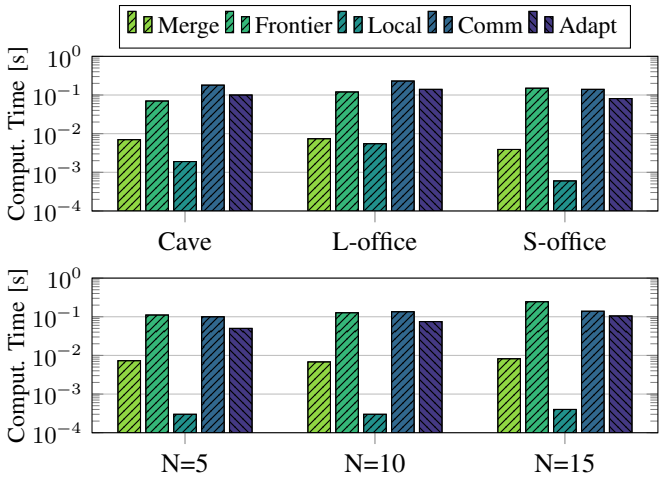

Fig. 16: The computation time of each step within the proposed method: for three scenarios (**Top**) and with different fleet sizes in the cave scenario (**Bottom**).

(II) **Prioritized region in** Q_1. Regarding the Q_1 requests where prioritized regions are specified online, tests are conducted for the large office environment with 3 robot number and $T_\mathrm{h} = 160s$. A small number of robots is adopted as this type of requests is useful when there are not enough robots to explore every branche simultaneously. Specifically, the first operator specifies the prioritized region located at the bottom-left (Region 1) at the first return event; while the second operator chooses top-left (Region 2), as shown in Fig. 14. In the first case, robot 0 returns at $134.3s$ and receives the request to explore region 1 first, which is then forwarded to robot 1 and 2 at $175.9s$ and $207.4s$. Then, their updated local plans take into account the prioritized region, which results in communication points near that region. Thus, instead of exploring the left branch first, all the robots first turn to the bottom branch and explore the prioritized region at $340.8s$. In contrast for the second case, the result is known to three robots at $131.1s$, $172.2s$ and $203.0s$ respectively. Different from the first case, with the updated plans one robot heads for region 2, one robot returns, and the other explores the bottom-left branch as one robot is enough to explore region 2 at $267.6s$. Note that in both cases, it is ensured that there is at least one robot assigned to explore the prioritized region, even when there are numerous frontiers and only a small number of robots. Last but not least, region 2 is finally explored at $520.8s$ for the first case, while region 1 is explored at $501.6s$ for the second case.

(III) **Dynamic movement of operator in** Q_2. Regarding the Q_2 requests where the operator moves to a desired location, the large office is tested with 6 robots and $T_\mathrm{h} = 160s$. The results are summarized in Fig. 7 and Fig. 13. It can be seen that the latency constraint is satisfied at all time even when the operator moves dynamically, with the maximum latency being $146.2s$. Compared with the case when the operator is static, the complete map is obtained at round $900s$. As shown in Fig. 7, the right-bottom area has been fully explored at $136s$, while the left part remains unknown. The operator requests to move to the left part, of which the feasible region is shown in purple and the intermediate goal point is marked by the red star. Thus, the operator moves left, then followed by another request to move to the bottom left part at $552s$. It can be noted that the maximum latency in the dynamic case changes more irregularly, as shown in Fig. 13. This is because the robots can communicate with the operator unexpectedly, yielding a decrease in latency. Notably, the second movement is essential to increase the explored area from 70% to 100%.

(IV) **Different human behaviors**. Last but not least, it is worth investigating how the proposed method reacts to *different* human behaviors. Particularly, consider the dynamic movement of the operator in the cave environment with 5 robots and $T_\mathrm{h} = 100s$. Fig. 15 summarizes the results of three different human behaviors: two operators choose different intermediate waypoints within the feasible region; and a random movement is also tested for comparison. Compared with only 59% coverage in the static case, the full map is obtained by both operators via 3 consecutive movements, at $512s$ and $481s$, respectively. Although two operators move in different paths, it is interesting to observe that they both are inclined to move towards the unexplored area, which is beneficial for subsequent exploration. Moreover, it can be seen that the whole map is obtained at $490s$ even when the operator moves randomly, however with more movements (6 vs. 3). This can be explained by the definition of the feasible region, which indeed serves as a guidance for the operator. In fact, a new position is feasible only if it is closer to communication points than the current position. Since the communication

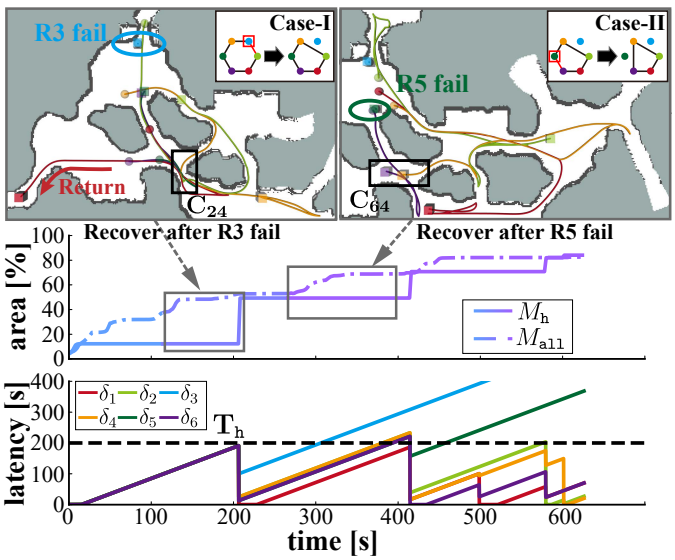

Fig. 17: Results for failure detection and recovery: robot trajectories (**Top**) after the failures of robots 3 (in blue) and 5 (in green), with new meeting points for recovery (black square), and the new communication topology (upper-right corner); the progress of exploration (**Middle**); the evolution of local latency for each robot (**Bottom**).

points are usually near the frontiers, the operator is more likely to move towards the unexplored area when moving in the feasible region. In other words, the proposed method remains effective under different human behaviors.

3) Complexity and Scalability Analysis: Due to its fully-distributed nature, the computation time of the proposed method mainly consists of two parts: (I) coordination during pairwise intermittent communication, including map merging, frontier generation, local plan optimization, and selection of next communication point; (II) local plan adaptation during local exploration when a frontier is reached. To begin with, the computation time of each step above is analyzed in different scenarios, where four robots are deployed in three scenarios from Fig. 10. As summarized in Fig. 16, both steps of map merging and local optimization take less than $0.01s$, while the generation of frontiers takes on average around $0.1s$, which is also the major cause of computation in the plan adaptation. Moreover, the selection of communication points takes around $0.1s$ due to the iterative search process as detailed in Alg. 1. Thus, the overall computation time of each meeting event is less than $0.5s$ on average, which is well suited for the event-based coordination. Second, the same analysis is performed for different fleet sizes, i.e., 5, 10 and 15 in the cave scenario. It can be seen from Fig. 16 that the computation time of different steps almost remains the same when robot number increases, which aligns with the distributed architecture. This indicates that the proposed method holds promise for potential application to an even larger fleet of robots.

4) Robot Failure: As described in Sec. III-D5, the proposed method can detect and recover from potential robot failures during execution. To validate this, consider the cave scenario

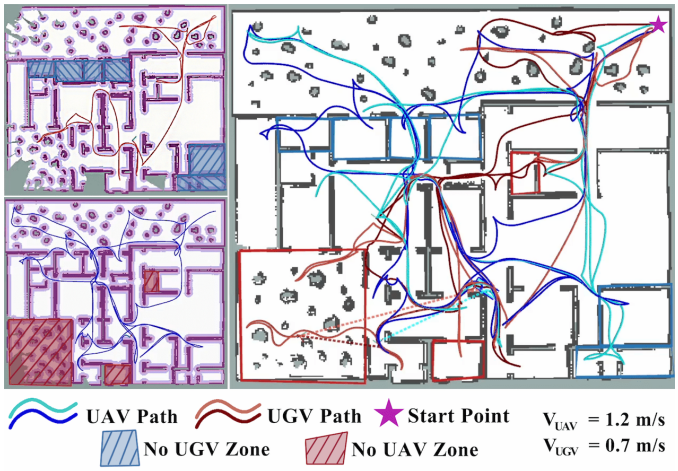

Fig. 18: Collaborative exploration via a heterogeneous fleet of 2 UAVs and 2 UGVs. **Left**: Trajectory and navigation costmap of one UGV (Top) and one UAV (Bottom); **Right**: Final trajectories of all robots within the explored map.

with 6 robots and $T_h = 200s$, of which two robots fail consecutively online. Particularly, the parameters T^-_{max} and T^+_{max} are set to $20s$ and $40s$, respectively. The final trajectories, explored area, and the latency of each robot over time are shown in Fig. 17. Robot 3 fails at $t = 110s$ *before* meeting with robot 2, which belongs to Case-I in Fig. 17. This failure is detected by robot 2 at $t = 171.8s$ after waiting for $20s$. Then, robot 2 directly communicates with the succeeding robot of 3, namely robot 4, thus the topology is adjusted by excluding robot 3. Furthermore, robot 5 fails at $t = 263s$ before communicating with robot 6, which belongs to Case-II in Fig. 17. Consequently, robot 6 detects the failure at $t = 305s$, after which it calculates a meeting point p_{64}, and then departs for robot 1. After three rounds of local communication, this information is propagated to robot 2, 3 and finally 4. Robot 4 then heads for p_{64} and meets with robot 6 at $t = 398s$, after which the recovery is completed. Note that the communication topology has reduced to a ring of three nodes, excluding robots 3 and 5. The overall exploration task is accomplished at $t = 624s$, with only a slight decrease in efficiency. Lastly, it is worth noting that the latency of functional robots may slightly exceed the threshold due to additional waiting time for failure detection, which is difficult to predict beforehand. Nonetheless, the latency constraint is satisfied again once the topology is recovered through subsequent coordination.

5) Heterogeneous Fleets: The proposed method is applied to a fleet of 2 UAVs and 2 UGVs with a maximum velocity of $1.2m/s$ and $0.7m/s$, respectively, to explore a large building complex with some forest area around. There are regions that are only accessible by UAVs (e.g., rooms with open windows and closed doors, marked by blue), and regions that are only accessible by UGVs (marked by red). The operator and all robots start from the top-right corner of the map, and T_h is set to $200s$. It can be seen that the UAVs have longer trajectories than UGVs ($(493m, 446m)$ vs. $(316m, 373m)$), and explore more area, due to their higher velocity. Moreover,

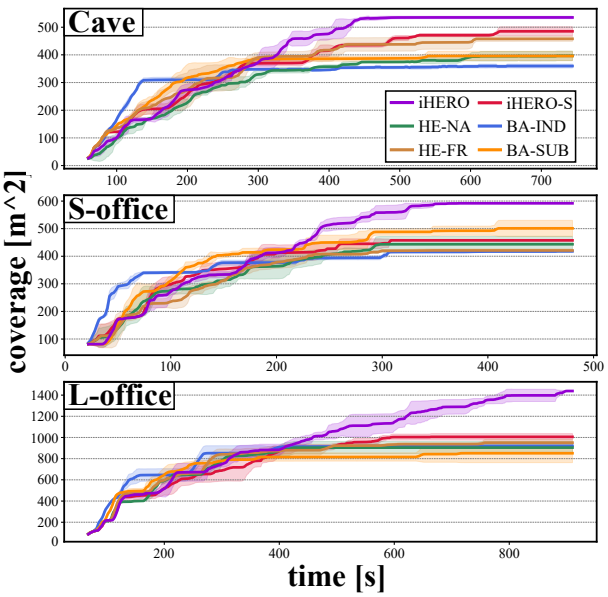

Fig. 19: Exploration progress of all methods under all three scenarios over three runs.

it is interesting to note that UAVs and UGVs exhibit different patterns of exploration due to different traversability. More specifically, after all robots have entered the building, the UAVs directly navigate through the top-left *windows* to explore the forest area, while it takes longer for the UGVs to reach the same area through the top-right door. Consequently, UAVs are assigned more frontiers to explore in the forest area, and UGVs mainly focus on the interior of the building. In other words, different capabilities of robots are fully utilized during exploration. Lastly, the operator moves into the building at $t = 371s$, and the whole map is explored at $t = 1080s$.

C. Comparisons

To further validate the effectiveness of the proposed framework (as **iHERO**), a quantitative comparison is conducted against **four** baselines:

(i) **BA-IND**, which is based on the distributed exploration framework in Burgard et al. [2], where each robot explores independently without active communication. To ensure the latency constraint, a modification is added such that each robot returns to the operator before its latency is predicted to be exceeding the bound;

(ii) **BA-SUB**, which is based on da Silva et al. [6], where the robots are divided into several subgroups, and only the robots in the same subgroup can communicate. Similar to BA-IND, at least one robot in each subgroup returns to the operator when the latency constraint is predicted to be exceeded;

(iii) **HE-FR**, where a fixed robot in the fleet is chosen to return to the operator, i.e., the operator is assigned a fixed neighbor in the communication topology;

(iv) **HE-NA**, where the local plan adaptation described in Sec. III-D3 is disabled. Thus, the plans generated during communication are strictly followed.

TABLE I: Comparison of baselines across three scenarios.

Scene	Method	Cover. Area[%]	Return Events[#]	Last Update[s]	Efficiency[m^2/s]	Online Interact.
Cave	**iHERO**	**100**	**4.3**	**453**	**0.82**	**Yes**
	iHERO-S	91.5	4.7	638	0.75	No
	HE-NA	74.8	5.7	600	0.61	No
	HE-FR	86.4	6.0	604	0.70	No
	BA-IND	67.9	16.0	428	0.55	No
	BA-SUB	74.7	10.7	536	0.61	No
S-office	**iHERO**	**100**	**3.7**	**326**	**1.69**	**Yes**
	iHERO-S	77.3	5.3	306	1.31	No
	HE-NA	74.9	5.7	290	1.27	No
	HE-FR	71.1	11.7	286	1.20	No
	BA-IND	70.6	16.0	304	1.19	No
	BA-SUB	84.7	10.7	408	1.43	No
L-office	**iHERO**	**100**	**5.3**	**907**	**1.58**	**Yes**
	iHERO-S	69.5	5.7	594	1.11	No
	HE-NA	62.5	6.3	484	0.99	No
	HE-FR	65.7	6.3	502	1.05	No
	BA-IND	63.6	16.0	355	1.01	No
	BA-SUB	62.3	13.7	402	0.94	No

Note that the operator has a request to move to the center of the explored area after the first return event, which is only supported with iHERO. Thus, an additional baseline is added as **iHERO-S**, where the operator is *not* allowed to move. The compared metrics are the covered area, the number of return events, the time of last update, the exploration efficiency (measured as the ratio between explored area and mission time), and whether online interactions are allowed.

In total 4 robots are deployed in the scenarios of large office, small office and the cave, with $T_{\mathrm{h}} = 150s$ in all cases, for which 3 tests are conducted for each method. As summarized in Table I and Fig. 19, the static method iHERO-S achieves the highest coverage and the smallest number of return events, among all baselines that do not support online interactions. Fig. 19 shows that although BA-IND has a high efficiency initially, the operator receives almost no new map after $150s$ as each robot has reached its maximum range given the latency. Similar phenomenon holds for HE-FR and BA-SUB at $400s$, which however have more return events in total than iHERO-S ($6, 10.7$ vs. 4.7 for the cave environment; $6.3, 13.7$ vs. 5.7 for the large office environment). However, choosing a fixed robot to return in HE-FR neglects the ongoing progress of exploration, while BA-SUB suffers from the lack of data flow between the subgroups, yielding even a lower efficiency. Moreover, the necessity of online adaptation is apparent as HE-NA has a much lower efficiency and a higher number of return events, compared with iHERO-S. In contrast, the naive method BA-IND has the lowest explored rate and highest number of return events among all methods, which validates the importance of inter-robot communication. Last but not least, as shown in Table I, iHERO is the **only** method that (i) achieves 100% coverage across all three scenarios; (ii) requires

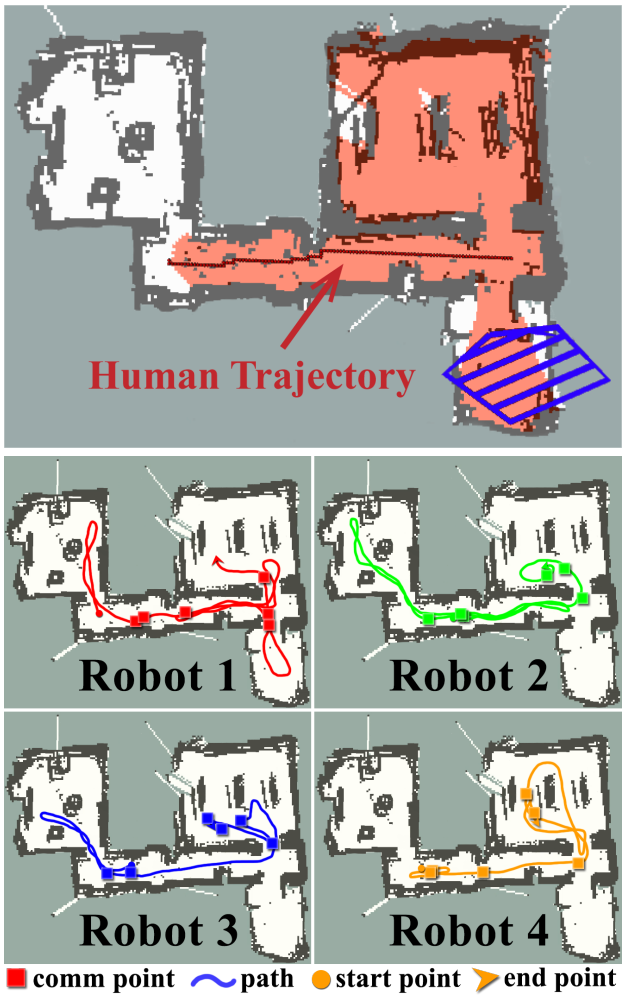

Fig. 20: Results from the hardware experiment. **Top**: the final map obtained by operator, including the prioritized region as Q_1 requests (blue polygon), and the dynamic movement of the operator as Q_2 requests to the right end, within the allowed region (in red); **Bottom**: the robot trajectories and communication points (colored squares).

the least number of return events than all baselines; (iii) has the highest efficiency over all baselines across all scenarios; (iv) supports online interactions such as Q_0, Q_1, Q_2 requests. This is also apparent by comparing the time of last update, i.e., the static methods can not generate new map updates after a short period for all scenarios. For instance, the method BA-IND can not explore new area after $355s$ in the scenario of large office, similarly for BA-SUB and HE-NA. Consequently, iHERO has a significant increase in efficiency: almost 56% compared with BA-IND and 68% with BA-SUB. This signifies the importance of dynamic interaction between the operator and the robotic fleet during collaborative exploration.

D. Hardware Experiments

1) System setup: As shown in Fig. 1, the hardware experiment deploys four differential-driven "LIMO" ground robots to explore an office environment. Each robot is equipped with an EAI XL2 LiDAR of range $10m$ and a NVIDIA Jetson Nano, which are responsible for running SLAM and navigation algorithms. Inter-robot communications follow the LOS constraints during runtime. The visualization and logging is on a laptop with Intel Core i7-1280P CPU, while the operator interacts with the robotic fleet through an Android tablet. The tested environment is of size $25m \times 20m$, composed of three rooms connected to a long corridor. The operator is an *expert* in robotics and familiar with the application of multi-robot collaborative exploration. A brief tutorial was offered before the mission, regarding what these requests are and how they can be specified on the tablet via a GUI as shown in Fig. 1. The operator is initially located at the start of corridor, without direct sight to the whole environment. The maximum speed of all robots is set to $0.3m/s$, and the latency $T_{\mathrm{h}} = 120s$.

TABLE II: Comparison of baselines in hardware experiments.

Method	Cover. Area[%]	Return Events[#]	Last Update[s]	Efficiency[m^2/s]	Online Interact.
iHERO	**100**	**3**	**484**	**0.40**	**Yes**
iHERO-S	93.5	4	518	0.37	No
HE-NA	88.1	6	482	0.35	No
HE-FR	91.6	4	581	0.37	No
BA-IND	85.2	10	440	0.34	No
BA-SUB	92.3	8	536	0.37	No

2) Results: Snapshots during hardware experiments are shown in Fig. 1, where the robots switch among different modes: exploration, intermittent communication and return to the operator. It is interesting to observe that different from simulations where a global view is available in the simulator, the operator can only obtain updates of the fleet status via the return events, e.g., which robot has explored which part and their current status. The final complete map and the robot trajectories are shown in Fig. 20. The latency constraint is satisfied at all time with the maximum latency being $102s$. Two different requests are handled during the exploration process: (i) Since the only remaining unknown parts are in the right area at $313s$, the operator sends a Q_1 request to prioritize the bottom-right room first (as shown in the blue polygon). Consequently, the robots react and the bottom-right room is fully explored at $433s$; (ii) the operator requests to move to the end of corridor at $440s$, for which the computed feasible region is shown in red. Then, the operator moves to the closest point in the feasible region to the desired location. Afterwards, the complete map is obtained at $484s$. It is worth emphasizing that due to actuation uncertainty and collision avoidance during navigation, the predicted arrival time of the robots when optimizing local plans is not always accurate. Thus, both the proposed intermittent communication and the online adaptation are essential to prevent the propagation of delayed arrival in most cases, thus ensuring the exploration efficiency and the latency constraint. Similar baselines are also tested in the hardware experiments, with results shown in Table II. They show that the proposed method has the potential

to be deployed to even larger fleets for real-world applications outside the lab. Detailed videos of the hardware experiments can be found in the supplementary material.

V. Conclusion

This work tackles the practical issues that arise during the deployment of multi-robot system for collaborative exploration, i.e., (i) the communication within the robotic fleet and between the fleet and the operator is restricted to close-range local communications; (ii) the operator requires a timely update of the exploration process online; (iii) the operator may specify prioritized regions to explore, or move dynamically within the explored area. An interactive human-oriented online coordination framework for collaborative exploration and supervision under scarce communication (iHERO) has been proposed. It builds upon the distributed protocol of intermittent communication, and accommodates explicitly these issues above as online requests $Q_{0,1,2}$ from the operator. It has been proven theoretically that the latency constraints in Q_0 are ensured at all time, while the $Q_{1,2}$ requests are satisfied online. Moreover, extensive numerical simulations are conducted across various scenarios with different robotic fleets, which validate that the proposed framework is the only method that supports online interaction with the operator, and achieves the highest performance w.r.t. the maximum area covered, the least number of return events, the overall exploration efficiency. Last but not least, a human-in-the-loop hardware demonstration is conducted where requests are specified online via the provided graphical interface, which further shows its potential applicability to real-world scenarios.

(i) As mentioned in Remark 3, the practical concerns of unaligned coordinate system and uncertainties during SLAM and map merging are not addressed in this work, which can be explored in future work; (ii) As discussed in Sec. III-D2, the communication topology $\mathcal{G}$ can be an arbitrary connected graph and set to a ring topology by default. However, as shown in the numerical experiments, such topology can have a significant impact on the exploration progress, which is hard to decide beforehand as the environment is unknown. Thus, an adaptive algorithm that changes the communication topology online according to the topological property of the explored area would be preferred, which is part of our ongoing work; (iii) Besides exploration, the robotic fleets often need to perform other tasks to interact with environment, such as inspection and manipulation, which however requires *consistent* and *high-bandwidth* communication (rather than intermittent). In this case, the other robots may need to temporarily stop exploration and serve as relays to form a chain of reliable communication. This leads to a combinatorial scheduling and assignment problem for unknown environments, which is part of our ongoing work; (iv) Lastly, although the method is proposed for general communication models, the numerical results are obtained under the simple model of LOS with a bounded range. Future work would involve more practical models depending on the hardware platform and applications.

Acknowledgement

This work was supported by the National Key Reasearch and Development Program of China under grant 2023YFB4706500; the National Natural Science Foundation of China (NSFC) under grants 62203017, U2241214, T2121002; and the Fundamental Reasearch Funds for the central universities.

References

[1] Torsten Andre, Daniel Neuhold, and Christian Bettstetter. Coordinated multi-robot exploration: Out of the box packages for ros. In *IEEE Globecom Workshops*, pages 1457–1462. IEEE, 2014.

[2] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E Schneider. Coordinated multi-robot exploration. *IEEE Transactions on robotics*, 21(3):376–386, 2005.

[3] Kyle Cesare, Ryan Skeele, Soo-Hyun Yoo, Yawei Zhang, and Geoffrey Hollinger. Multi-uav exploration with limited communication and battery. In *IEEE international conference on robotics and automation (ICRA)*, pages 2230–2235, 2015.

[4] Rafael Gonçalves Colares and Luiz Chaimowicz. The next frontier: Combining information gain and distance cost for decentralized multi-robot exploration. In *Annual ACM Symposium on Applied Computing*, pages 268–274, 2016.

[5] Micael Santos Couceiro. An overview of swarm robotics for search and rescue applications. *Artificial Intelligence: Concepts, Methodologies, Tools, and Applications*, pages 1522–1561, 2017.

[6] Alysson Ribeiro da Silva, Luiz Chaimowicz, Vijay Kumar, Thales Costa Silva, and Ani Hsieh. Communication-constrained multi-robot exploration with intermittent rendezvous, 2023.

[7] Yuman Gao, Yingjian Wang, Xingguang Zhong, Tiankai Yang, Mingyang Wang, Zhixiong Xu, Yongchao Wang, Yi Lin, Chao Xu, and Fei Gao. Meeting-merging-mission: A multi-robot coordinate framework for large-scale communication-limited exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13700–13707, 2022.

[8] Meng Guo and Dimos V Dimarogonas. Multi-agent plan reconfiguration under local ltl specifications. *The International Journal of Robotics Research*, 34(2):218–235, 2015.

[9] Meng Guo and Michael M Zavlanos. Multirobot data gathering under buffer constraints and intermittent communication. *IEEE Transactions on Robotics*, 34(4):1082–1097, 2018.

[10] Dirk Holz, Nicola Basilico, Francesco Amigoni, and Sven Behnke. Evaluating the efficiency of frontier-based exploration strategies. In *International Symposium on Robotics*, pages 1–8. VDE, 2010.

[11] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient

probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 34:189–206, 2013.

[12] Ahmed Hussein, Mohamed Adel, Mohamed Bakr, Omar M Shehata, and Alaa Khamis. Multi-robot task allocation for search and rescue missions. In *Journal of Physics: Conference Series*, volume 570, page 052006, 2014.

[13] Yiannis Kantaros, Meng Guo, and Michael M Zavlanos. Temporal logic task planning and intermittent connectivity control of mobile robot networks. *IEEE Transactions on Automatic Control*, 64(10):4105–4120, 2019.

[14] Filip Klaesson, Petter Nilsson, Tiago Stegun Vaquero, Scott Tepsuporn, Aaron D Ames, and Richard M Murray. Planning and optimization for multi-robot planetary cave exploration under intermittent connectivity constraints. 2020.

[15] Pierre-Yves Lajoie, Benjamin Ramtoula, Yun Chang, Luca Carlone, and Giovanni Beltrame. Door-slam: Distributed, online, and outlier resilient slam for robotic teams. *IEEE Robotics and Automation Letters*, 5(2): 1656–1663, 2020.

[16] Yaroslav Marchukov and Luis Montano. Fast and scalable multi-robot deployment planning under connectivity constraints. In *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 1–7, 2019.

[17] Hans Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 116–121, 1985.

[18] Indraneel Patil, Rachel Zheng, Charvi Gupta, Jaekyung Song, Narendar Sriram, and Katia Sycara. Graph-based simultaneous coverage and exploration planning for fast multi-robot search. *arXiv preprint arXiv:2303.02259*, 2023.

[19] Yuanteng Pei, Matt W Mutka, and Ning Xi. Coordinated multi-robot real-time exploration with connectivity and bandwidth awareness. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5460–5465, 2010.

[20] Pavel Petráček, Vít Krátkỳ, Matěj Petrlík, Tomáš Báča, Radim Kratochvíl, and Martin Saska. Large-scale exploration of cave environments by unmanned aerial vehicles. *IEEE Robotics and Automation Letters*, 6(4):7596–7603, 2021.

[21] Martijn N Rooker and Andreas Birk. Multi-robot exploration under the constraints of wireless networking. *Control Engineering Practice*, 15(4):435–445, 2007.

[22] Maira Saboia, Lillian Clark, Vivek Thangavelu, Jeffrey A Edlund, Kyohei Otsu, Gustavo J Correa, Vivek Shankar Varadharajan, Angel Santamaria-Navarro, Thomas Touma, Amanda Bouman, et al. Achord: Communication-aware multi-robot coordination with intermittent connectivity. *IEEE Robotics and Automation Letters*, 7(4):10184–10191, 2022.

[23] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[24] Yulun Tian, Yun Chang, Fernando Herrera Arias, Carlos Nieto-Granda, Jonathan P How, and Luca Carlone. Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems. *IEEE Transactions on Robotics*, 38(4), 2022.

[25] Tiago Vaquero, Martina Troesch, and Steve Chien. An approach for autonomous multi-rover collaboration for mars cave exploration: Preliminary results. In *International Symposium on Artificial Intelligence, Robotics, and Automation in Space (SAIRAS)*, 2018.

[26] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 146–151, 1997.

[27] Brian Yamauchi. Decentralized coordination for multi-robot exploration. *Robotics and Autonomous Systems*, 29 (2-3):111–118, 1999.

[28] Boyu Zhou, Hao Xu, and Shaojie Shen. Racer: Rapid collaborative exploration with a decentralized multi-uav system. *IEEE Transactions on Robotics*, 2023.

APPENDIX

A. Proof of Lemma 1

Proof: To begin with, when $k = 0$, $\chi_0^{\mathrm{h}} = 0 > t_0 - T_{\mathrm{h}} = -T_{\mathrm{h}}$; when $k \geqslant 1$, $\chi_k^{\mathrm{h}} > \chi_{k-1}^{\mathrm{h}} \geqslant (t_k - T_{\mathrm{h}})$. Thus, $\chi_k^{\mathrm{h}} > (t_k - T_{\mathrm{h}})$ holds, $\forall k \geqslant 0$. It implies that the constraint (7) is fulfilled when $t \in \{t_k, k \geqslant 0\}$. Second, due to the non-decreasing monotonicity of t_n^{h}, it holds that $t - t_n^{\mathrm{h}}(t) \leqslant t - t_n^{\mathrm{h}}(t_k)$. Thus, the inequality $t - t_n^{\mathrm{h}}(t_k) \leqslant t - \chi_k^{\mathrm{h}} \leqslant t - (t_{k+1} - T_{\mathrm{h}}) \leqslant T_{\mathrm{h}}$ holds, yielding that the constraint (7) is fulfilled, $\forall t \in [t_k, t_{k+1}]$. This completes the proof. ■

B. Proof of Lemma 2

Proof: To begin with, if the while-loop in Lines 9-14 of Alg. 1 terminates before $\widetilde{\mathcal{F}}_{ij}$ becomes empty, the resulting solution is clearly feasible. If $\widetilde{\mathcal{F}}_{ij}$ becomes empty, consider two cases: (i) If a return event is not required, event $c_{ij}^{\star}$ derived in Line 3 is clearly a feasible solution; (ii) If a return event is required and robot j returns to the operator, it follows that:

$$\begin{aligned} &\max_{\ell=i,j} \left\{ t_\ell^{K_\ell} + T_{\mathtt{nav}}(p_\ell^{K_\ell}, p_{ij}^{\star}) \right\} + T_{\mathtt{nav}}(p_{ij}^{\star}, p_{\mathrm{h}}) \\ =& \max \left\{ t_i^{K_i} + T_{\mathtt{nav}}(p_i^{K_i}, p_{\mathrm{h}}),\, t_j^{K_j} \right\} \\ \leqslant& \max \left\{ T_{\mathtt{lim}}^{K_i},\, T_{\mathtt{lim}}^{K_j} \right\} \leqslant T_{\mathtt{lim}}, \end{aligned} \tag{18}$$

where $p_{ij}^{\star}$ is derived in (12); $p_j^{K_j} = p_{\mathrm{h}}$ as robot j returns to the operator; $t_j^{K_j}$ is the estimated arrival time of robot j at p_h; $T_{\mathtt{lim}}^{K_i}$ is the upper-bound for choosing $c_i^{K_i}$ (the same for j); and $T_{\mathtt{lim}}$ is the updated time limit after taking into account the return event of robot j, see Line 7. The first equality of (18) stems from the fact that $p_{ij}^{\star}$ belongs to the generated path from $p_i^{K_i}$ to p_{h}, while the first inequality is ensured by Alg. 1 for the previous meeting events. Moreover, the last inequality follows from the non-decreasing monotonicity

of $T_{\mathtt{lim}}$, which can be inferred from (10), (13) and (15). Note that inequality 18 still holds true for the first round, when there is no previous meeting event for robot i or j to ensure its first inequality. This is due to the fact that initially the robots are within the range of communication with operator, which means $t_i^{K_i} + T_{\mathtt{nav}}(p_i^{K_i}, p_{\mathrm{h}}) = 0$ if i has no previous meeting events, and $t_j^{K_j} = 0$ for the case of j. Thus, Alg. 1 always returns a feasible solution for all meeting events. ■

C. Proof of Theorem 1

Proof: Lemma 2 has ensured that the coordination during each pairwise communication is guaranteed to return a feasible solution. Thus, it remains to show that the two conditions of Proposition 1 are satisfied. Regarding the first condition, Alg. 1 ensures that all communication events satisfy the constraint (11), meaning that the time of the next return event is bounded by $t_{r+1} \leqslant \mathbf{min}_n\{\Omega_{ij}^{\mathrm{h}}[n]\} + T_{\mathrm{h}} \leqslant \chi_r^{\mathrm{h}} + T_{\mathrm{h}}$. The second condition is ensured by the cyclic communication topology and the pairwise coordination process described above. More specifically, assume that robot i returns to the operator before communicating with its succeeding robot j in $\mathcal{G}_{\mathrm{S}}$ during round r. Due to the cyclic communication topology, $\mathbf{argmin}_n\{T_r^{\mathrm{h}}[n]\} = j$. Considering the strategy of choosing the robot to return as described before, the robot that returns to base in round $r+1$ is either j or any other succeeding robots of j, which means that T_{r+1}^{h} should contain the information of robot i and j's meeting event. Without loss of generality, assume that robot j's succeeding robot is k, then it holds that $\mathbf{min}_n\{T_{r+1}^{\mathrm{h}}[n]\} \geqslant t_{ij} > t_{jk}^{-} = \mathbf{min}_n\{T_r^{\mathrm{h}}[n]\}$, where t_{jk}^{-} is the meeting time of communication event between robot j and k that happens before robot i returns, and t_{ij} is the meeting time of i and j after i returns, implying that the second condition is satisfied. And if the adopted topology is not a ring topology, consider a robot i communicating with robot j in c_{ij}, to decide their next event c'_{ij}. If i returns to the operator at t_i, then equation (11) implies that $t_i \leqslant t_i^{\mathrm{h}}(t_i) + T_{\mathrm{h}}$; while if i doesn't return, the strategy of determining return events ensures that $t - t_i^{\mathrm{h}}(t) < T_{\mathrm{h}}, \forall t \in [t_{ij}, t'_{ij}]$, where t_{ij} is the meeting time of i and j in c_{ij}, and t'_{ij} is the meeting time of i and j in c'_{ij}. Thus, time constraint is satisfied locally for each robot i, and hence the global constraint is satisfied for the whole fleet. This completes the proof. ■