IEEE Transactions on Field Robotics (T-FR) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-FR paper.

Deployment of an Aerial Multi-agent System for Automated Task Execution in Large-scale Underground Mining Environments

Niklas Dahlquist[1], Samuel Nordström, Nikolaos Stathoulopoulos, Björn Lindqvist, Akshit Saradagi and George Nikolakopoulos

***Abstract*— In this article, we present a framework for deploying aerial multi-agent systems in large-scale subterranean environments with minimal supporting infrastructure. The objective is to optimally and reactively execute routine inspection tasks, selected by a mine operator on-the-fly. The assignment of currently available tasks to the agents is accomplished through an auction-based system, where the agents bid for available tasks, which are used by a central auctioneer to optimally assign the tasks. A mobile Wi-Fi mesh supports inter-agent communication and bi-directional communication between agents and the task allocator, while the task execution is performed completely infrastructure-free. Given a task to be accomplished, reliable and modular agent behavior is synthesized by generating behavior trees from a pool of agent capabilities, using a back-chaining approach. The auction system is reactive and supports the addition of new tasks on-the-go, at any point through a user-friendly operator interface. The framework has been validated in a real underground mining environment using three aerial agents, with several inspection locations spread in an environment of almost 200 meters as a proof-of-concept. The scalability, fault tolerance, and the influence of agent initializations on the multi-agent architecture have been tested through complementary Gazebo simulations in a cave environment. The proposed framework can be utilized in a subterranean environment for missions involving rapid inspection, gas detection, distributed sensing and mapping etc. The proposed framework and its field deployment contribute towards furthering reliable automation in large-scale subterranean environments to offload both routine and dangerous tasks from human operators to autonomous aerial robots.**

I. Introduction

The use of autonomous robotic platforms in industrial production facilities is on the rise, both to increase profitability and to increase safety for human operators [1]. Specifically, in deep underground mining, where the fundamental risk of accidents is high, the mining industry is focusing on creating a safer environment for humans by deploying robotic systems to either execute dangerous tasks or verify the safety before authorizing human entry. Through efforts in the mining industry, human workers have already been moved to safer locations in several critical operations via, for instance, teleoperation of heavy machinery. For tasks that still require direct human involvement, it is crucial to track the status of the environment to minimize the risk before executing the necessary work. Currently, many measures are taken to reduce risks to humans in mining environments, such as tags carried by the operators to keep track of all personnel inside the subterranean environment, fixed sensors to monitor seismic activity, air quality, etc. In this setting, autonomous aerial robotic inspection offers a revolutionary alternative, through which routine large-scale status monitoring and continuous data collection can be performed efficiently with minimal human involvement. The mining areas that require inspection are vast in size (anywhere from 100 m to several kilometers) and present a challenging scenario for aerial robotic inspectors with limited battery and traversal speeds. This necessitates the transition to deploying autonomous multi-agent aerial robot systems for such large-scale operations [2].

The central problem in efficient and effective deployment of an aerial multi-robot system, in missions such as routine inspection, dangerous gas monitoring, and change detection [3], is that of coordinating task execution between the participating robotic agents. The coordinating entity/architecture is responsible for the optimal distribution of available tasks among participating agents, and it is expected to be reactive, in order to handle sudden changes in both the environment and the overall mission plan. When considering the execution of the assigned task itself, a large portion of the development effort is spent on designing robust local behaviors associated with a particular mission. Therefore, it is valuable to utilize a modular design to allow for easy integration of new tasks or new robotic platforms into the coordinating architecture. This article presents an auction-based coordination architecture for aerial multi-agent systems to optimally distribute inspection tasks among available agents and a modular approach for synthesizing agent behaviors for task execution.

Deploying multi-agent systems in subterranean mining environments usually implies that there exist limited communication possibilities, especially in certain areas, where it is necessary to design the overall architecture to be resilient to degraded performance. In this work, a big emphasis is laid on i) designing the local autonomy of the individual agents, to make sure that tasks can be executed independently even in the case of communication failure, and ii) designing the task allocation architecture, utilizing communication only for reactively allocating the available tasks, to enable large-scale missions in active underground mining environments and to achieve a predictable and well defined behavior in case of temporary communication failure.

The performance of the proposed architecture has been

[1] Niklas Dahlquist is the corresponding author of the article, nikdah@ltu.se. All authors are with Luleå University of Technology (LTU), Sweden. This work has been funded in part by the Swedish Department of Energy and LKAB through the Sustainable Underground Mining (SUM) Academy Programme project "Autonomous Drones for Underground Mining Operations".

(a) (b) (c) (d)

Fig. 1: The evaluation environment at a depth of over 500m in an active production area of the mine located in Kiruna, Sweden. (a) and (b) show the tunnel environment, with the aerial vehicles on the ground before a mission is initiated, (c) illustrates one of the many challenges for autonomy in a mining environment, flying in smoke-filled or dusty environments. Finally, in (d), a snapshot of an empty tunnel is presented to show the scale of the operating environment.

validated through the deployment of a three-agent aerial robotic system in a mining environment to execute an inspection mission. Being able to perform full-system experiments in real operational environments is a great opportunity. We would like to thank our mining industry collaborators at Luossavaara-Kiirunavaara AB (LKAB), who operate the largest underground iron ore mine in the world. The field demonstrations included in this manuscript were enabled through the Sustainable Underground Mining (SUM) Academy programme, which is funded by LKAB and the Swedish Energy Agency, and were executed at a depth of over 500m in an active production area of the mine located in Kiruna, Sweden.

A. Challenges for Multi-agent Deployment in Underground Mines

Deploying robotic systems in underground mines, or in subterranean environments in general, comes with a specific set of challenges [4] related to onboard perception and navigation systems (a few scenarios presented in Fig. 1). These types of environments are significantly perception-degraded; darkness is limiting the use of visual sensors, and large amounts of dust (or smoke in some use-cases) that can potentially corrupt or block LiDAR data. The environment geometry itself, consisting of narrow tunnels, significantly limits the field of view of onboard sensors, which, in combination with the self-similarity of the environment, pose significant challenges to the SLAM/localization system. Safe navigation and collision avoidance are also heavily stressed as the robot platform is constantly close to the tunnel walls or installed infrastructure, especially for aerial robots, where any interaction with the environment can lead to a crash – and sets definite requirements on any navigation stack in terms of robot safety. Moreover, the environment is highly dynamic, where heavy vehicles,

ore piles, and infrastructure are constantly moved around to facilitate the extraction operations, while the tunnels themselves expand and change over time. As such, both the navigation and control stack, as well as the overarching task coordination layer, must be reactive to changes in both the environment and current mission. For multi-agent coordination, there are also limitations on the available connectivity and communication infrastructure. A completely centralized system for both task coordination and task execution would require real-time perception/mapping data from all agents. In our proposed framework, this challenge of minimizing the required communication bandwidth is addressed by limiting the communication between agents and the coordinator to 1) broadcast available tasks (coordinator →agents), 2) share individual task costs (agents→coordinator, and 3) convey allocated tasks (coordinator→agents), which is later visualized in Fig. 2.

B. Related Works

1) Mining & Subterranean Robotics: While deploying aerial robots in underground mines is challenging, the inherent safety risks and environmental challenges (vast size, 3D terrain etc.) also present significant opportunities for developing resilient autonomy solutions. The survey [5] discusses how aerial robots (at the time mainly teleoperated) have been used in underground mines for gas concentration measurements [6], visual and thermal inspection of rock mass characteristics [7], inspection of the distribution of rock fragmentation (or muck piles) after blasting [8], and for providing visual inspection of unreachable areas [9]. Towards the vision of fully automating such inspection tasks in real mining environments using autonomous ground vehicles, there exists literature considering tasks such as air quality monitoring and detection of cracks in the tunnel walls [10], [11]. A large driving force in the recent developments and up-scaling of autonomous robotic missions in mining/subterranean environments can be attributed to the DARPA Subterranean Challenge [12], where teams deployed fleets of robots into fully unknown subterranean environments in the search-and-rescue context, which, in contrast to our work assumes no information about the environment and zero infrastructure for communication. The final round of the DARPA Subterranean Challenge required the competing teams to deploy teams of heterogeneous agents to explore an unknown environment and report artifacts during 60-minute missions utilizing expert human supervision. The highest scoring team [13] mainly utilized ground-based platforms to explore and only ground-based agents to identify and report the scoring artifacts. During that period, the two central problems of subterranean localization/SLAM [4], [14]–[16] and navigation [17] received special attention and the state-of-the-art was significantly extended. The challenge also saw the deployment of multi-robot systems [18] for collaborative mission execution. The robot collaboration was centered around multi-modal robots [19], in guiding robots not to explore overlapping areas through deployed communication nodes [20], multi-robot SLAM [21], and multi-robot task allocation [22], which is the most relevant in the context of our article. There are also several works in the literature where a single robot system is deployed for subterranean missions, such as [23], and these also serve as primary baselines for our work. In general, very few works exist in the academic literature that can showcase multi-robot collaborative deployment in the context of task allocation in real field mining environments. The framework presented in this manuscript not only deploys a novel task allocator and is demonstrated in a real field environment – it was also developed specifically to enable a non-expert user to easily interact with the system through an easy-to-use GUI, which allows a single worker to operate the whole robotic fleet.

The main difference between our approach and the one presented in [22] stems from the different use cases. Our work is focused on routine inspection and integrating aerial agents to be a part of the mining environment itself, while requiring communication infrastructure to enable monitoring the fleet at all times, in contrast to focusing on exploring and searching for objects in a completely unknown environment [22]. This sets our work apart, as we utilize the communication infrastructure to enable a centralized operator to reactively assign available tasks (in this work, the inspection of certain locations) to autonomous agents, which execute the tasks by visiting the inspection locations. This allows us to have a more predictable coordination strategy without needing to rely on a fully decentralized task allocation strategy. The focus of this article is not on addressing the challenges involved in distributed multi-robot coordination, but on proposing and experimentally demonstrating the utility of an aerial multi-robot system for safely and reliably executing routine tasks in mining environments and thereby pushing for their adoption in the near future.

Optimal multi-agent task allocation can be achieved through many approaches. The survey [24], on multi-robot systems, identifies four components in a typical complex multi-agent mission: 1) task decomposition, 2) coalition formation, 3) task allocation, and 4) task execution/planning and control. This work focuses on designing a multi-agent coordination framework that focuses on the third and fourth components and assumes discrete tasks that do not require decomposition and can be accomplished by a single agent. In this work, we employ a market-inspired auction-based task allocation scheme to optimally assign available tasks to agents (for the reasons elaborated in section III-B). The works [25]–[27] also utilize auctioning systems for multi-agent coordination, where the tasks are distributed among agents based on their bids. However, the validation of the auctioning systems is demonstrated through limited, laboratory-scale experiments. It is also worth noting that, none of the works cited above automate all four components of multi-agent coordination identified in [24]. Our work in [28] presented a similar reactive auction-based task allocation strategy combined with behavior trees for tracking task execution, but without any field experiments for verifying the performance of the system and to ensure the transferability of the overall framework to a real-world large-scale scenario.

In addition, a major difference between the current work and [28] is that in the current work, an aerial multi-agent system is deployed in a distributed way, with the individual agents having a full autonomy stack that is capable of executing tasks by navigating in a large-scale mining environment.

Recent works on multi-drone systems have demonstrated deployment in real-world outdoor environments for tasks such as mapping and exploration. In [29], the authors present a multi-drone platform for real-life deployments and a multitude of applications, including mapping and aerial exploration. Similarly, [30] designs a multi-drone system for forestry applications, focusing on autonomous navigation and sensing to enable large-scale environmental monitoring of forest areas, with possible application in multiple similar scenarios. However, these studies do not consider the unique challenges related to field deployment in harsh underground mining environments where extra strain is put on the system and the onboard autonomy, and the specific requirements posed in this work for the intended use case of large-scale routine inspections.

2) Multi-robot Task Execution Planning and Allocation: To enable autonomous robots to execute complex tasks in diverse environments, it is essential to develop a flexible mechanism that utilizes a pool of modular robot behaviors to generate an optimized sequence of behaviors to accomplish the assigned task, while also considering the safety constraints imposed by the surroundings. Generating task plans involves the process of determining the sequence of actions and resources required to complete a task efficiently. Task plans can be generated dynamically based on current conditions and constraints, as in works such as [31], where task planning is accomplished considering only partial knowledge of the environment for the scenario where a robot needs to navigate and find specific objects. Classical methods such as Petri nets, decision trees, and state machines [32] have been widely utilized in task planning to model and formalize complex behaviors. These methods provide a structured approach to represent and analyze the behavior of automated systems. The main drawback of these methods is the lack of modularity and a natural way to include reactivity. The authors of [33] note that many of the currently available task planning methods result in a static plan of actions to solve the desired problem without the necessary flexibility to react to unforeseen events.

Behavior trees have recently demonstrated a fair amount of success in deploying robotic systems in uncertain and dynamic environments due to their modularity and reactivity. Behavior trees are shown to generalize other common control structures, such as state machines and decision trees [34], and require less programming effort to develop and maintain due to their inherent modularity [35]. Recently, a method for generating reactive behavior trees for a robotic platform, based on a set of available actions and conditions, was presented in [36]. In this article, we adopt the theoretical method presented in [36] with several adjustments to enable the generation of a complete behavior tree for aerial robots prior to deployment. The sequential reasoning behind the synthesis of behavior trees aids in verifying and guaranteeing that the task planning would accomplish a given task. We present a large-scale experimental verification by deploying the method for aerial platforms which accomplish tasks in an underground mining environment, as part of a multi-agent system.

C. Contributions

In this work, we develop a flexible aerial multi-agent autonomy framework specifically suited for enabling task execution in underground mining environments. The contributions of this article in relation to the available literature can now be stated as follows.

- We present a reactive aerial multi-agent solution for simultaneous execution of routine inspection tasks in an active underground mine, which is motivated by the need in the mining industry for a safer and more efficient subterranean workspace. We experimentally demonstrate the merits of the proposed solution through large-scale field deployment in an underground mine that has no supporting infrastructure for multi-agent robotics.
- The proposed methodology comprises of two main parts: i) global coordination orchestrated by a central auction-based task allocation system and ii) local autonomy centered around automated behavior tree generation, where a back-chaining approach is used to synthesize behavior trees from a pool of primitive agent capabilities. We design a set of primitive capabilities required for autonomous aerial agents and utilize the back-chaining approach to enable inspection-style tasks.
- Reliable integration of local autonomy and task allocation layers to enable reactive multi-agent coordination through: i) an easy-to-use operator interface, to allow for user-specific tasks to be added on-the-go, ii) deployment of a mobile Wi-Fi mesh to enable inter-agent communication and agent participation in the auctioning system, and iii) integration of the supporting elements of the autonomy stack, such as risk-aware path planning, global relocalization in a point-cloud, and NMPC/APF-based tracking and collision avoidance.

In addition, we include a discussion on structuring and executing large-scale multi-agent experiments in a mine, along with practical concerns and lessons learnt from our experience in deploying a team of aerial agents in an active underground mining environment in collaboration with the mining company Luossavaara-Kiirunavaara AB (LKAB).

II. Problem Overview

In subterranean environments, deployment of an aerial multi-agent system on a large scale, with respect to both the size of the environment and the number of agents, poses several challenges. Three of the most significant challenges that lead to the research problem addressed in this article are listed below.

Reactive task allocation: The foremost issue is ensuring efficient and optimal task allocation to the agents while being

able to reactively re-distribute the available tasks amongst the agents, due to the uncertainty posed by a dynamic and unpredictable environment. Such a framework should be able to quickly adapt to a priori unknown factors, such as addition of new tasks, changes in the operating environment (unexpected obstacles), or agent failures during task execution.

Modular and flexible control architecture: In addition to having a flexible task allocation framework, an efficient control architecture is essential to coordinate the actions of agents seamlessly to perform different kinds of available tasks. The need to have the capability to, on-the-fly, utilize the modular behaviors of individual agents to compose and plan the execution of complex tasks is apparent.

Versatile and user-friendly operator interface: To enable the deployment of such an autonomous multi-agent system in an industrial production environment, it is vital to have a versatile and easy-to-use operator interface to supervise and orchestrate the deployment. The operator interface could be used to introduce new tasks on-the-fly, command specific agents directly, set constraints based on the changing environment etc. Through the interface, the operators would possess valuable situational awareness/domain knowledge and would have the ability to interact and inform the task allocation process, both for specifying the actual tasks to perform and the task execution order. In short, a single operator at the surface control center, potentially kilometers away, should easily be able to control a fleet of robotic agents through the user interface, which acts as the face of the proposed task assignment framework and robust mission-specific local autonomy.

In summary, the research problem that this article looks to tackle can be stated as follows. The goal is to design a reactive task allocation framework that is suitable for unpredictable underground environments, such as deep underground mines, that lends a high level of autonomy to the agents while also allowing operators to interact and introduce tasks to the system on-the-go. To facilitate this, an efficient way of automatically structuring the agent autonomy has to be designed, which at the agent level utilizes simple and modular agent capabilities to enable agents to perform many different types of tasks; and at the level of the multi-agent system enables easy integration of new tasks and capabilities into the system.

III. Multi-agent Autonomy in SubT Environments

This section presents three sub-components of the overall multi-agent autonomy and the expectations from the three sub-components when deployed in subterranean environments.

A. Local Autonomy

To deploy multi-agent systems in a constantly changing and uncertain environment, it is necessary to give a large degree of autonomy to the individual agents. In certain scenarios, the agents are designed to perform a discrete set of modular actions/behaviors, and this set of actions can vary from agent to agent in a heterogeneous multi-agent system. A mechanism or methodology for optimally composing and sequencing the available modular actions into a control architecture for an agent to execute complex tasks is crucial. Both the methodology and the resulting architecture are expected to have several features, including:

1) *Flexibility*: Different tasks may require different sets of actions or behaviors. By having a set of available actions for every agent, it is possible to mix the agents to create teams tailored to specific mission requirements. This flexibility allows for the adaptation of robotic systems to a wide range of tasks and environments.
2) *Reusability*: Developing new actions from scratch for each specific task can be time-consuming and resource-intensive. A library of available actions allows developers to reuse and repurpose components that have already been created and tested. This can significantly reduce development time and costs to include new types of tasks.
3) *Modularity*: A library of actions encourages a modular approach to system design. Each action can be developed and tested independently, making it easier to debug and maintain the overall system. Additionally, modular architectures are inherently more scalable, allowing for easier inclusion of new functionality and sensing capabilities.
4) *Robustness*: By having a library of well-tested actions, developers can leverage existing implementations to create more robust control architectures. Actions that have been thoroughly tested and validated are less likely to contain bugs or unexpected behaviors, resulting in more reliable robotic systems.

Behavior trees are well known for satisfying all the features mentioned above [34], [35]. The individual actions are collected as behaviors and are therefore inherently flexible, reusable and modular. The increased robustness stems from the ability to reuse certain well-tested pieces of code, even on multiple different platforms. In the context of the multi-agent subterranean mission scenario considered in this work and in the software architecture to be proposed in section IV, behavior trees will be a central part of the local autonomy of the agents.

B. Task Coordination

A reactive task allocation framework is necessary to handle the uncertainty in the environment, specifically in the subterranean mining environment scenario considered in this work, and to allow for efficient insertion of tasks during mission execution. For handling multi-agent systems, such a framework is expected to be flexible to include different types of tasks, allow for heterogeneous agents to participate, dynamically adapt based on a changing environment, scale well with a large number of agents, and give a considerable amount of autonomy to the agents to execute the tasks. The market-based task allocation architecture has been a classic approach [25] for enabling the deployment of multi-agent

systems in such environments. The features that motivate the use of this multi-agent coordination approach include:

1) *Flexibility*: The adaptable nature of auction-based task allocation enables real-time and on-the-fly adaptation to changing factors such as fluctuations in the set of available tasks, agent availability, and dynamically changing environments. This flexibility enables the system to quickly respond and react to new information and optimally reassign tasks to agents. This is especially important in the multi-agent mission considered in this scenario, where an operator can, at any point in time, add new tasks, and the system needs to quickly adjust and optimally reallocate tasks.
2) *Fairness*: Auction-based methods provide a transparent and unbiased way of allocating tasks, ensuring that all participants have an equal chance to bid for, and get allocated to, the available tasks, based on their capabilities and current knowledge. The fairness enables trust between the agents and allows for agents with different capabilities to compete within the same system.
3) *Scalability*: Separation of optimal assignment (by the central auctioneer) and cost estimation (bids computed by the agents) into two separate processes makes an auction-based task allocation framework inherently scalable and capable of handling both large number of agents and large number of available tasks without suffering from performance degradation. This scalability makes it suitable for large-scale deployment where reactivity is a priority, since it might be infeasible to plan the optimal task allocation over a long horizon.
4) *Decentralization*: By allowing agents to independently generate bids and affect the auction process through the bids, auction-based frameworks promote distributed decision-making. Having a partially decentralized decision process can lead to more robust and resilient systems that are less likely to fail due to single points of failure.
5) *Optimality*: Auction-based frameworks encourage agents to provide their best offers, leading to optimized task allocations that maximize overall system performance. Where reactivity is the main priority, an auction-based system can give an optimal solution over a short horizon, given the current information about the system and the environment.

C. Environment and Agent Tasks

Before presenting the framework for multi-agent coordination proposed in this work in the next section, we present a precise definition of the environment where the framework is expected to be deployed.

The overall multi-agent system is expected to be deployed in a deep underground mine, where the environment is both challenging and unique, with narrow and confined tunnels. The type of tasks that the multi-agent aerial system is expected to execute in the field deployment are "inspection" style tasks such as: gas monitoring, rock-face inspection, etc. The main target is to reactively allocate the tasks in a flexible and optimized manner. To enable autonomous inspection, especially gas monitoring after blasting, where the ground could be partially covered by rocks or other debris, aerial agents are required for robust operation independent of the state of the ground in the mining environment. The choice of tasks is motivated by the need of the mining industry to optimize the rock-wall inspection when developing new tunnels, monitoring air quality after blasts for increased security, and so on. Given this scenario, the algorithms and software modules developed in this work are tailored for solving such an autonomous mission. The local autonomy of the agents is designed using automatically generated behavior trees due to the flexible and reactive nature of behavior trees and since it allows for rapid integration of additional tasks and robotic platforms without changing large parts of the developed software. A similar perspective guides the design of the auction-based task allocation to maximize the autonomy given to the agents in the resulting framework.

1) Ease of Use for Operators: For autonomous robotics to be accepted by the mining industrial community and to enable frequent deployments in the field, an absolute requirement is to create a user-friendly interface for the operators, which allows them to supervise and interact with the framework, without the need to be trained in the details of the underlying framework. In this work, the task allocation framework is designed so that the only input that is required is the type of task to be executed and the location where the task has to be performed. This allows for an easy-to-use interface, where an operator can choose a location within a three-dimensional map where an inspection task needs to be carried out.

IV. Methodology

A. Automated Behavior Tree Synthesis for Agent Autonomy

A behavior tree can be represented as a directed acyclic graph $\mathrm{BT} = (\mathcal{N}, \mathcal{V})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{V}$ defines the set of connections forming the behavior tree, examples of which can be found in Figures 7, 8 and 9. The leaf nodes are called *execution nodes* and the internal nodes are called *control nodes*. A behavior tree is executed by ticking the root node, this "tick" is passed down by the *control nodes* until an *execution node* receives it. A detailed description of the concept of behavior tree and the commonly used *control nodes* can be found in [34].

Consider the case where a library of available actions $\mathcal{L} = \{\mathcal{A}_i, {\mathcal{A}^{\mathrm{Pre}}}_{i,j}, {\mathcal{A}^{\mathrm{Post}}}_{i,k}\}$ that a specific robotic platform can perform is known. The library $\mathcal{L}$ consists of a set of actions $\mathcal{A}_i$, where $i \in \mathbb{N}$ is the number of available actions, a set of necessary conditions ${\mathcal{A}^{\mathrm{Pre}}}_{i,j}$, where $j \in \mathbb{N}$ represents the conditions that needs to be satisfied before action $\mathcal{A}_i$ can be performed. The set of post conditions ${\mathcal{A}^{\mathrm{Post}}}_{i,k}$, where $k \in \mathbb{N}$ represents the conditions that the action $\mathcal{A}_i$ will satisfy after the action is executed. It is assumed that there exist no cyclic dependencies between the pre-conditions and the post-conditions, meaning that we do not consider the case where an action would invalidate one or more of the conditions that are necessary to execute that said action. In addition, it

is also assumed that the actions $\mathcal{A}_i$ correspond to behaviors that can be executed as *execution nodes* and the pre- and post conditions can be implemented as *condition nodes*.

1) Behavior Tree Synthesis: The principle of back-chaining, initially applied to behavior trees for the first time in [36], stems from the idea that a goal condition is specified and then, in a backward fashion, the necessary pre-conditions and actions are added. The Algorithms 1 and 2 detail how the algorithm is implemented, similar to [36] but with the key difference that all conditions are expanded to make sure that it is feasible to achieve the desired condition before execution. In simple terms, the algorithms generate the task-specific behavior tree by iteratively expanding every condition by adding actions that satisfy that said condition until there are no more conditions to be expanded. This allows tasks to be defined as any condition to be achieved, and as long as the agent's capabilities allow for it, a platform-specific behavior tree can be synthesized to execute that specific task.

2) Algorithm Steps in Detail: i) Initialize the behavior tree (Algorithm 1 line 2): The behavior tree is initialized by adding the final condition associated with completing the requested task. The behavior tree is then expanded until all actions required for satisfying this condition are added.

ii) Run until all necessary conditions are expanded (Algorithm 1 line 3 – 6): This loop is executed until all conditions in the resulting behavior tree have been expanded to satisfy all its preconditions.

iii) Expand condition (Algorithm 2): This Algorithm replaces a condition node C with a sub-tree including the necessary actions to satisfy said condition. It works as follows: First an action $\mathcal{A}$ is selected that will satisfy the condition C; second all pre-conditions $[\mathcal{A}^{\mathrm{Pre}}{}_1, \ldots, \mathcal{A}^{\mathrm{Pre}}{}_n]$ of $\mathcal{A}$ are added in a sequence before the action $\mathcal{A}$ and this sub-tree is called T_1; Then, T_1 is added in a fallback together with the original condition C; Finally, the condition C is replaced in the behavior tree with the generated sub-tree T_2.

B. Auction-based Optimal Task Allocation

In order to present the inner workings of the proposed task allocation architecture, this section begins by defining the necessary notations. In a multi-agent setting, the set of available tasks, at a specific time instant t, is denoted by $\mathfrak{T}_t = \{T_1, T_2, \ldots, T_{n_t}\}$, where $n_t \in \mathbb{N}$ is the number of tasks available at time t. The tasks are considered to have one stage, for example "inspect a location", "pick-up an object" etc. A team of agents $\mathfrak{R} = \{R_1, R_2, \ldots, R_{n_a}\}$ are available for executing the tasks, where $n_a \in \mathbb{N}$ is the number of agents. We consider the setting where the tasks and their arrival times are not known a priori, i.e the set $\mathfrak{T}_t$ is a time-varying set, into which new tasks are added as they arrive and completed tasks are removed.

In this work, a market-inspired auction system, visually summarized in Fig. 2, is employed for task allocation and it comprises of three stages: 1) Announcement stage: The auctioning system announces the available tasks $\mathfrak{T}_t$ to the agents. 2) Bidding stage: The agents $R_k \in \mathfrak{R}$ compute the costs ($c_{k,j}$) for accomplishing tasks $T_j \in \mathfrak{T}_t$ and send bids to the auctioning system. 3) Task allocation stage: The task allocator solves a combinatorial optimization problem to decide how the tasks $\mathfrak{T}_t$ should be distributed among the agents $\mathfrak{R}$. The allocated tasks are then announced to the agents. Note that the central coordinator/allocator is an external entity and not one of the aerial agents themselves. Since the cost computations are performed locally by the agents, the central allocator has no need for the global information of the multi-agent system, and this significantly reduces the complexity of the combinatorial task allocation problem and minimizes the communication requirements. Thanks to the modularity and simplicity of the auctioning system, the three constituent steps can be run at a high rate. This also keeps the communication overheads low by requiring the transmission of only: 1) the set of available tasks $\mathfrak{T}$, 2) the bids from the agents $c_{k,j}$ and 3) the allocated tasks x^* respectively during the three stages.

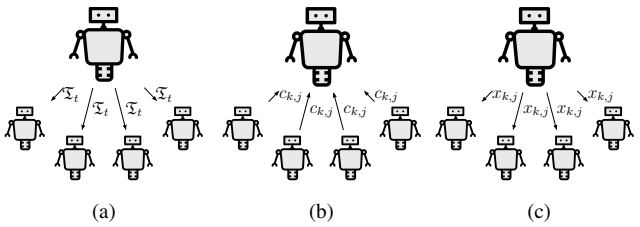


Fig. 2: An illustration of the different stages of the auction system. (a) Announcement stage. Available tasks $\mathfrak{T}_t$ are announced to all agents. (b) Bidding stage. Agents calculate costs $c_{k,j}$ and return them to the auctioneer. (c) Task allocation stage. The auction system optimizes the task assignment and broadcasts $x_{k,j}$, which is set to one if agent k is assigned to task j and zero otherwise.

1) Optimization Formulation: To decide the optimal task allocation, the centralized task allocator solves a combinatorial optimization problem, specifically the following linear integer program (LIP):

$$\begin{aligned} x^* = &\arg\max_{x_{k,j}} \sum_{(k,j)\in E} \rho_{k,j} \cdot x_{k,j} \\ \text{s.t.} \quad &\sum_{\{k|R_k\in\mathfrak{R}\}} x_{k,j} \leq 1,\ \forall j \in \{1,\ldots,n_t\} \\ &\sum_{\{j|T_j\in\mathfrak{T}_t\}} x_{k,j} \leq 1,\ \forall k \in \{1,\ldots,n_a\}, \end{aligned} \tag{1}$$

where $x_{k,j} \in \{0,1\}$ is an assignment variable indicating if agent R_k is assigned to task T_j and the assignment matrix $x^\star \in \{0,1\}^{n_a \times n_t}$ defines the optimal distribution of tasks $\mathfrak{T}_t$ among the agents $\mathfrak{R}$. $\rho_{k,j} \in \mathbb{R}$ represents the 'profit' for agent R_k to complete task T_j. E is the set of two-tuples representing the edges of a bipartite graph between the set of agents and the set of tasks. The profits $\rho_{k,j}$ are found by transforming the costs $c_{k,j}$ in a way that reverses the order. This is done to make sure that the optimization problem assigns as many agents as possible to the available tasks. Here, it should be noted that the applicability of the optimization formulation (1) can be easily expanded to more complex scenarios by including factors such as:

Algorithm 1 Pseudocode for generating the fully expanded behavior tree using the back-chaining approach.

Input: C_{start} : Starting condition
Output: BT : Final behavior tree

function GENERATEBEHAVIORTREE(C_{start})
  BT $\leftarrow C_{\text{start}}$ ▷ Add initial condition
  while $ConditionsToExpand \notin \varnothing$ **do**
    $C_{\text{expand}} \leftarrow getConditionToExpand(\text{BT})$
    BT $\leftarrow Expand(\text{BT}, C_{\text{expand}})$
  end while
  return BT
end function

Algorithm 2 Pseudocode for expanding a single condition.

Input: BT : Behavior tree with condition to expand
  C : Condition to expand
Output: BT : The expanded behavior tree

function EXPAND(BT, C)
  $\mathcal{A} \leftarrow FindActionThatSolvesCondition(C)$
  $[\mathcal{A}^{\text{Pre}}{}_1, \ldots, \mathcal{A}^{\text{Pre}}{}_n] \leftarrow GetPreConditionsToAction(\mathcal{A})$
  $T_1 \leftarrow Sequence([A_1^{\text{Pre}}, \ldots, A_n^{\text{Pre}}], \mathcal{A})$
  $T_2 \leftarrow Fallback(C, T_1)$
  BT $\leftarrow Replace(\text{BT}, C, T_2)$
  return BT
end function

relative importance of tasks, time prioritization of tasks, and balancing the execution between different classes of tasks. This is done by scaling the additive terms appropriately or by adding additive terms and constraints corresponding to each factor, as shown in [37].

2) Cost Estimation: The estimation of the costs $c_{k,j}$, for the available tasks $\mathfrak{T}_t$ is performed locally by every agent R_k. The cost $c_{k,j}$ submitted by an agent, for completing the inspection tasks considered in this work, is based on the risk-aware path (elaborated in subsection IV-C.2) from the current agent location to the inspection location. More specifically, the total distance of the risk-aware path from the current location of agent R_k to the inspection location of task T_j is considered as the total cost $c_{k,j}$ to complete that task. In situations where a team of heterogeneous agents would bid for the same tasks, this cost could, for example, be scaled to reflect the difference in traversing speed of the various agent types.

C. Agent Autonomy

In this work, the local autonomy of the agents is designed to be able to operate independently, without relying on centralized infrastructure. Each agent manages its operation and reacts to changes in the environment to ensure resilient and robust operation. This approach allows the task execution to be handled locally in a completely infrastructure-free environment, only utilizing communication infrastructure to enable communication with the central auctioneer responsible for optimal task allocation. Fig. 3 shows a high-level description of how the local autonomy communicates with the coordination layer and Fig. 4 details the modules of the local autonomy running on every agent.

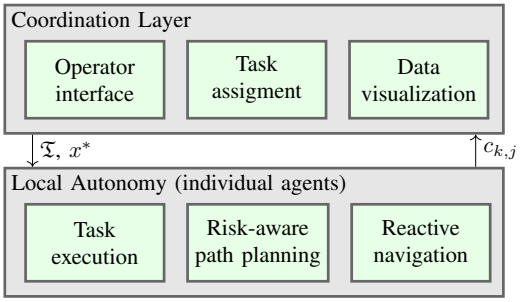


Fig. 3: An overview of the operation of the overall multi-agent coordination architecture. The figure highlights the key components of the coordination and local autonomy layers. The coordination layer requires only the estimated task execution costs $c_{k,j}$ to determine the optimal allocations x^* from the pool of available tasks $\mathfrak{T}$. Fig. 4 illustrates how every agent operates locally in greater detail.

1) Global and Local Localization: To localize the agents, we utilize the publicly available LiDAR-Inertia Odometry framework, FAST-LIO [38], which provides odometry at 10Hz by fusing LiDAR and IMU data. It is optimized for computational efficiency and robustness, making it ideal for UAVs with limited payload capacity. FAST-LIO uses a tightly-coupled, iterated Extended Kalman Filter to integrate LiDAR feature points with IMU data, facilitating navigation in high-speed, noisy, and cluttered environments. The method for calculating the Kalman gain is designed to be efficient, as it depends on the state dimension rather than the

measurement dimension, thereby lowering the computational burden.

In a multi-agent system, effective and efficient task execution as well as waypoint navigation require localization within a common coordinate frame. To align the local odometry from FAST-LIO with the world coordinate frame $\mathcal{W}$, an initialization process transforms the local robot frame $\mathcal{R}$ to the world frame $\mathcal{W}$ through a homogeneous rigid transformation of the special Euclidean group, $T : \mathcal{R} \to \mathcal{W}$. The extension of FAST-LIO[1] requires an initial guess T_0 to perform a two-step ICP (Iterative Closest Point) registration and computation of the refined transformation T before starting the odometry estimation. Operators can manually provide the initial guess for each agent, especially if missions begin from unique starting points. Alternatively, a fixed starting position can be set to a predetermined initial location, though this introduces unnecessary constraints and reduces the system's autonomy.

To address this challenge, frameworks similar to our previous work, 3DEG [39] can be utilized. Given a point cloud map, 3DEG can estimate the transformation matrix $T \in SE(3)$ that aligns the current robot frame $\mathcal{R}$ with the global map frame $\mathcal{W}$. Using a point cloud map $M_{pcd} \in \mathbb{R}^3$ and its corresponding trajectory $T_r \in \mathbb{R}^3$, 3DEG determines the rigid transformation matrix T from a single LiDAR scan. The process starts by partitioning the map and deriving the set of n descriptive vectors $Q = \{\vec{q}_1, \vec{q}_2, \ldots, \vec{q}_n\}$, where each element $\vec{q}_i \in \mathbb{R}^{64}$ and $W = \{\vec{w}_1, \vec{w}_2, \ldots, \vec{w}_n\}$, where each element $\vec{w}_i \in \mathbb{R}^{64}$. The vectors $\vec{q}_i$ encode location-specific information, while the vectors $\vec{w}_i$ encode orientation-specific data, enabling the estimation of the yaw difference between point cloud scans. Next, a sample from the LiDAR is used to extract the current descriptor $\vec{q}_t$. This descriptor is then used to query the map's vectors, identifying the nearest neighbor $\vec{q}_t^*$ through the following optimization routine:

$$\vec{q}_i^* = \underset{\vec{q}_i \in Q}{\arg\min}\, f(\vec{q}_i, \vec{q}_t), \tag{2}$$

where the cost function $f : \mathbb{R}^{64} \times \mathbb{R}^{64} \to \mathbb{R}$ captures the similarity between the descriptive vectors, in this work, the euclidean distance between the two vectors is used to determine similarity. The index i of the nearest neighbor $\vec{q}_i^*$ in the set Q is used to find the corresponding vector $\vec{w}_i^*$ from W, which are together used to construct the initial guess transform T_0, which is then passed to FAST-LIO. Finally, FAST-LIO uses this estimate to perform the two-step ICP algorithm, refining the alignment between the robot frame and the current map, ensuring precise localization.

2) Risk-aware Path Planning: To plan a path from the current location of an agent to the location where an assigned task has to be executed, the path planner D^*_+ [40] is used. D^*_+ is a risk-aware path planner that considers unknown areas and areas close to obstacles and walls as risk areas. For such areas, in computing the traversal cost c, additional costs are assigned to account for the risks, over the standard distance

[1] https://github.com/HViktorTsoi/FAST_LIO_LOCALIZATION

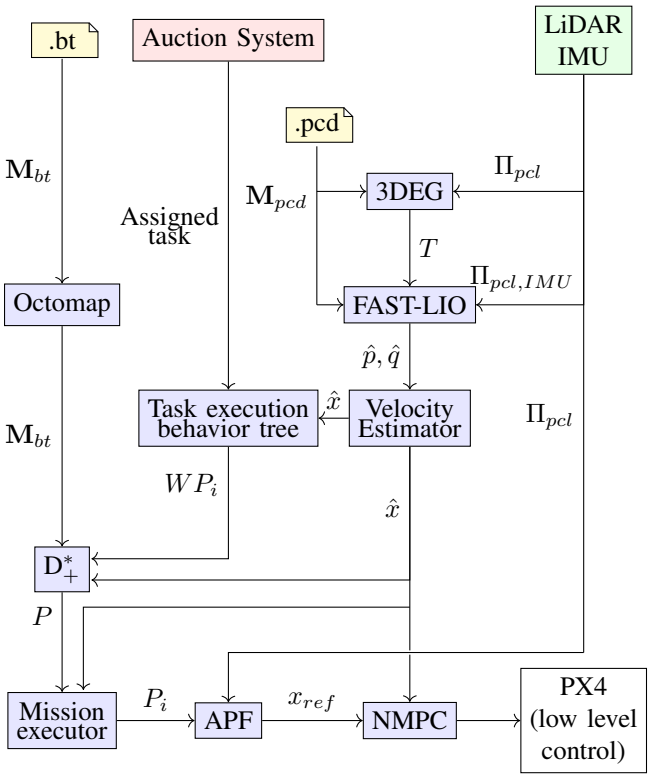


Fig. 4: A block diagram describing the local autonomy that runs on every agent, to executes a task assigned by the auctioning system (shown in red). The blue boxes are the software that constitutes the autonomy stack. The sensors used by the agents are marked in green. The maps that are loaded from files are marked in yellow. The white components are entities that take inputs from the agent-autonomy stack. PX4 is the onboard low-level controller that is directly controlling the aerial agents.

costs/heuristics. D^*_+ is based on the occupancy voxel map provided by OctoMap [41], and the resulting path plan P is a list of voxels that a UAV must sequentially visit, to move from point 'A' to point 'B'. D^*_+ plans the cheapest path P between two points in the map, by minimizing the sum of the traversal cost along a path P, i.e, $\sum_{c_i \in P} c_i$. The traversal cost c for each voxel is calculated as follows:

$$c_r = \begin{cases} \frac{c_u}{d+1} & \text{if } d < r \\ 0 & \text{else} \end{cases} \tag{3}$$

$$c = \begin{cases} c_r + c_d & \text{if the voxel is free} \\ c_u + c_r + c_d & \text{if the voxel status is unknown} \end{cases} \tag{4}$$

where d is the distance of the voxel from the closest occupied voxel and r is the radius in which risk is evaluated. c_d is the standard cost that is proportional to the travel distance or length of the path. The cost for unknown voxels c_u and the risk cost c_r are part of the total cost to traverse through that voxel, which is denoted as c. Equation (4) is constructed to create a risk gradient, meaning that the further away a robot is from an occupied cell the lower the risk it is. The division of the cost c_u by $d+1$ accounts for the considered notion of risk. By changing the balance between the different costs, different behaviors can be promoted. For example, a small c_u encourages more exploratory paths, while a large c_d

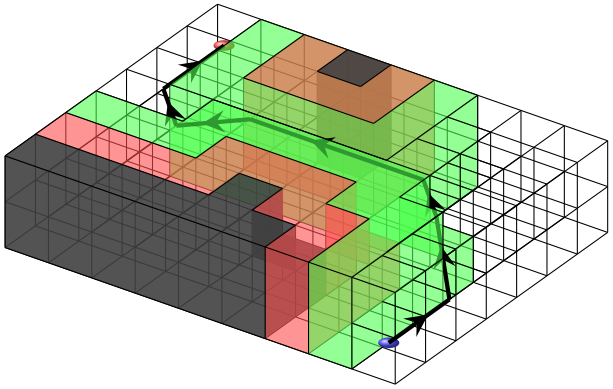


Fig. 5: An illustration of how D$^*_+$ plans a path from the blue ball to the red ball, by taking the risk layer into account. The red voxels are assigned higher risk, the green voxels are assigned lower risk, and black voxels are occupied.

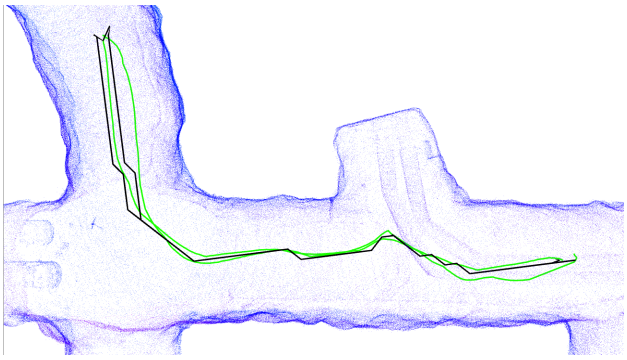


Fig. 6: A comparison between the planned path (black line) and the path the aerial agent followed (green line) during navigation. Using the local avoidance and path tracking, P is followed closely with only minor deviation while staying at a safe distance from any obstacle.

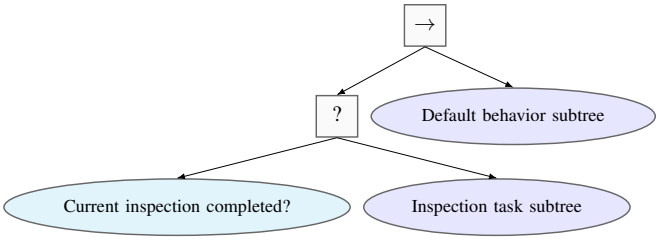


Fig. 7: The behavior tree structure used to switch between inspection tasks and a default behavior. The red nodes are later expanded as sub-trees based on the generated, back-chained, task-specific behavior trees.

shifts priority to promote shorter paths. The path P provided by D$^*_+$, is the shortest and safest path, as all three costs c_r, c_d, and c_u are considered in planning a path. It is the combination of all those costs that makes P safe for a robot to traverse. An illustrative case of D$^*_+$ planning is shown in Fig. 5. The result from the deployment of D$^*_+$ in a subterranean field experiment is presented in Fig. 6, where both the planned path and the path tracked by an aerial robot are illustrated, showing that D$^*_+$ planning chooses the safest route whenever possible.

In this article, the agents estimate the cost for executing an available task by computing the length of the D$^*_+$ path P that safely takes a UAV agent from its current location 'A' to the target location 'B' associated with an available task. The costs computed by the agents are sent as bids to the central auction-based task allocation system for incorporating in the optimization problem (1) in section IV-B.1. D$^*_+$ is chosen because of its capability to reliably produce safe-to-navigate paths for aerial agents.

3) Reactive Path Tracking & Local Avoidance: Each individual agent uses a reactive local navigation kit [42] to track D$^*_+$ paths to target inspection locations. The kit is composed of a high-performance nonlinear model predictive controller (NMPC) combined with a fully reactive Artificial Potential Field (APF). The APF uses raw 3D LiDAR scans to generate repulsive forces (avoidance maneuvers) from obstacles and walls. The kit acts as an additional safety layer to ensure robot safety while traversing the tunnels to the inspection locations, resulting in small insignificant deviation from the planned path as can be seen in Fig. 6.

4) Behavior Tree Structure: To enable the use of the back-chaining principle to switch between generated task-specific behavior trees, the behavior tree structure in Fig. 7 is used, specifically for performing inspection tasks with a default (when not allocated to a task) behavior of returning to a home location.

5) Task Specific Behavior Trees: The library of available actions $\mathcal{L}$ utilized to generate the required task-specific behavior trees, as detailed in Algorithms 1 and 2, can be seen in Table I. The resulting task-specific behavior trees utilized in the inspection mission, one for executing a task and one for defining the agents' default behavior, are shown in Fig. 8 and Fig. 9.

V. Mission Description and Preparations

A. Hardware

The three aerial platforms used for field demonstrations are custom-built drones designed by the Robotics & AI Group at Luleå University of Technology, Sweden. The design and a schematic figure of the platforms are shown in Fig. 10. Most of the autonomy components of the proposed framework partly rely on information from the 3D LiDAR sensor since the drone is required to navigate and localize in dark environments. In the development process, we evaluated multiple LiDAR systems and settled on Ouster OS1 32-beam, due to its robustness when operating in dusty environments commonly encountered in underground mines.

The UAV also carries a Pixhawk Cube FCU (Flight Control Unit) that handles all low-level control, with its internal IMU (Inertial Measurement Unit) also providing acceleration and roll/pitch data to the Lidar-Inertial localization framework. All computation is handled by an onboard Intel NUC. Not shown in the figure is a GoPro camera that was added to the front of the UAV to provide visual imagery during inspection missions. The camera was selected simply for its large internal storage, low weight and ease of use.

TABLE I
THE LIBRARY $\mathcal{L}$ OF ACTIONS AND PRE- AND POST-CONDITIONS USED TO GENERATE TASK-SPECIFIC BEHAVIOR TREES IN THE EVALUATION SCENARIOS.

Action, $\mathcal{A}_i$	Precondition, ${\mathcal{A}^{\text{Pre}}}_{i,j}$	Postcondition, ${\mathcal{A}^{\text{Post}}}_{i,k}$
Set home location	-	Has home location
Arm	-	Is armed
Set flight mode OFFBOARD	-	In OFFBOARD mode
Takeoff	Has home location, Is armed, In OFFBOARD mode	Is Flying
Follow path	Is flying	At goal point
Update path	-	Has path

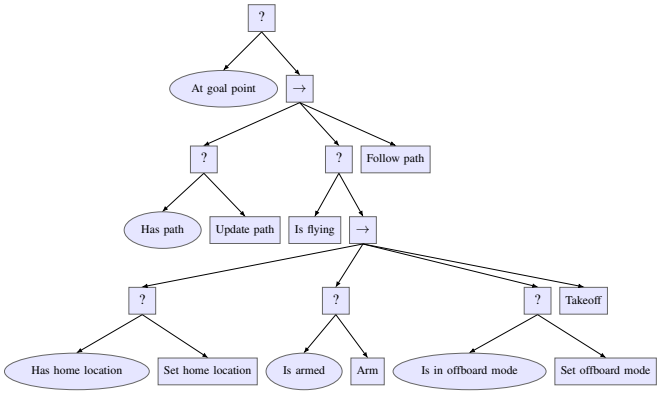


Fig. 8: The behavior tree generated through the back-chaining approach, for executing inspection tasks. The Algorithm 1 was used to generate the behavior tree, with 'At goal point' as the required condition.

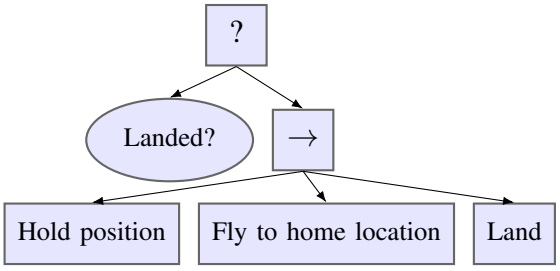
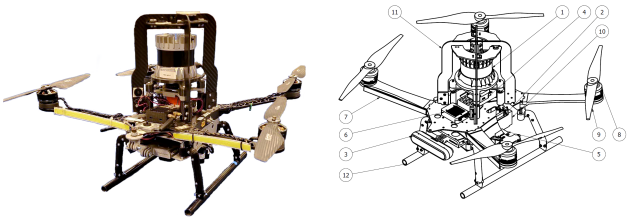


Fig. 9: Generated behavior tree for executing the default behavior. When no task is allocated for a specified time, the agent executes the default behavior of returning to its home location to land and wait for a new task to be allocated.



(a) (b)

Fig. 10: The platform used during the field deployment. In the main mission, three of the shown aerial vehicles were deployed to collaboratively execute the multi-agent mission. (a) shows an image of the real platform and (b) shows a schematic drawing with the following key parts specified: 1- Ouster OS1 3D LiDAR, 2- Intel NUC, 3- Intel Realsense D455, 4- Pixhawk Cube Flight Controller, 5- Garmin singlebeam LiDAR, 6- Telemetry module, 7- LED strips, 8- T-motor MN3508 kV700, 9- 12.5 in Propellers, 10- Battery, 11- Roll cage, 12- Landing gear.

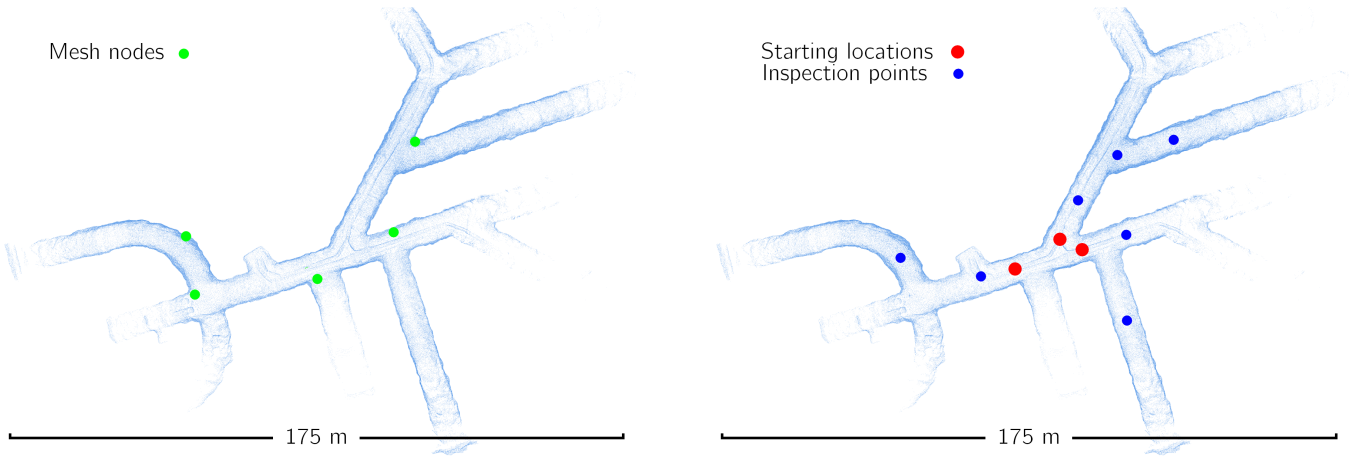
B. Mission Set-up

1) Preparation Phase: Before the mission begins, thorough preparation is essential, with respect to setting up the UAV agents, communication infrastructure, and deployment protocols. This involves ensuring that all agents are equipped with the required sensors, such as a 3D LiDAR for localization and scanning and cameras for visual feedback and visual inspection, and communication equipment. A portable, battery-powered, Wi-Fi mesh, detailed in section V-C, is set up to establish communication between the agents and the auctioning system for reactive task allocation among the agents during the mission and to allow for adding new tasks on-the-fly. Fig. 13 shows an image of one of the five Wi-Fi nodes deployed in the subterranean environment. Each Wi-Fi node consists of: 1) the communication hardware, the Wi-Fi 6 enabled access point U6 Pro from Ubiquiti Inc. 2) a battery pack providing 48 V power over Ethernet and 3) a tripod raising the node to an efficient height, ensuring that the battery pack is kept away from the wet/muddy floor, and good communication is established between the nodes.

The execution of large-scale robotics experiments requires a team effort, and setting routines for coordination and communication protocols is important. A safety briefing is conducted for the entire team, detailing the mission objectives, communication protocols, and emergency procedures, to ensure that everyone is aware of their roles and responsibilities and how to react in case of an accident.

2) Deployment Phase: The deployment phase begins with the setup of a launching area, which serves as the control center during the mission. This is typically established at a safe, stable location of the mine. From this area, the mission is started and the aerial agents are launched. Additionally, initial system checks are performed to ensure that the mission can be launched.

To enable the use of multiple agents in a shared environment, a mapping session is performed prior to the inspection phase. The 3D map can be constructed either with a handheld device or by an autonomous agent [43] through an exploration mission. Once the point cloud map is constructed and loaded onto all agents, the agents can be re-localized within it, in order to have a common coordinate frame, either manually by the operator or automatically as described in subsection IV-C.1 utilizing the pre-scanned global map. Then, the desired inspection locations, including the type of inspections that need to be performed during this specific



(a) (b)

Fig. 11: Data showing the scale of the underground environment where the multi-agent deployment was executed. (a) Shows the map and the locations of the deployed communication nodes, and (b) illustrates the starting position and the locations for the inspection tasks that were executed during the mission. The map is the same as was used for localization and path planning during the mission.

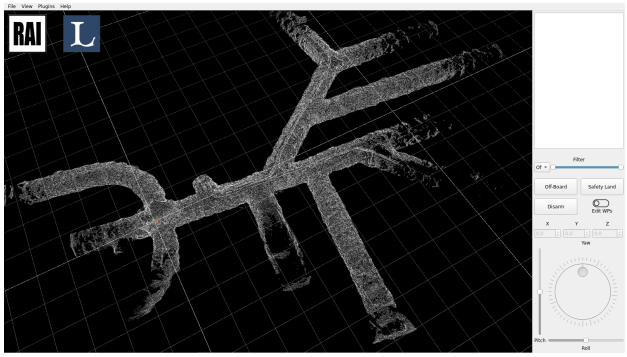


Fig. 12: A view of the operator's graphical interface where inspection tasks were added to initiate and interact with the mission. The interface allows the operator to add and edit 3D locations, which are sent to the task allocation system as requests for autonomous inspection.

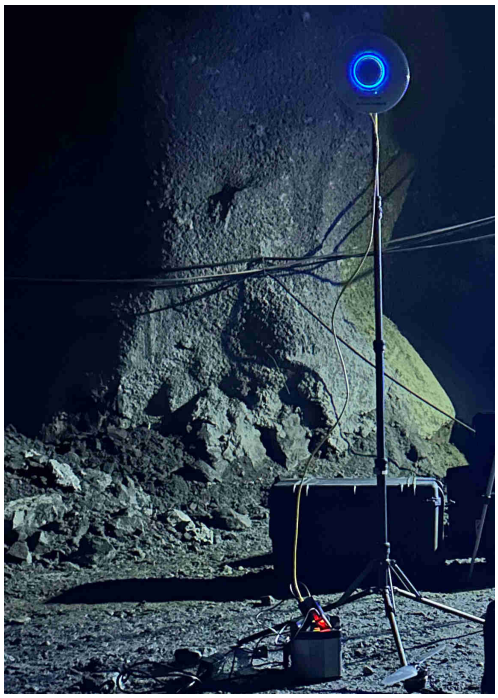


Fig. 13: One of the nodes in the mobile Wi-Fi mesh used to provide communication for the task coordination. The Ubiquiti U6 Pro is mounted on a tripod and connected to a battery to allow for easy and quick deployment in the underground environment.

mission, are chosen and the inspection mission is ready to be carried out by the aerial agents.

3) Inspection Phase: During the inspection phase, the aerial agents execute the assigned inspection tasks by navigating through the environment to the target locations. The path to the target locations is generated using the D^*_+ risk-aware path planner (section IV-C.2) and the paths are tracked using NMPC, with APF providing reactive local avoidance and an additional safety layer (section IV-C.3). As the inspection tasks are carried out, the agents transmit the gathered data back to the control center to be viewed in real time. This enables real-time mission adjustments, addition of new inspection tasks on-the-fly, and removal of inspection tasks that are no longer deemed necessary. Throughout the field deployments presented in this work, the control center hosted the auctioning system and the user interface, using which as operator could select inspection locations and monitor the mission execution.

The mission concludes with a debriefing to review the mission's outcome, discuss any challenges faced, and identify areas for improvement. By following this structured approach, safe and efficient inspection missions are assured, and continuous improvements to future inspection missions can be achieved.

C. Limited Infrastructure

Each agent has its own autonomy stack on board, in the sense that it can operate without any external infrastructure. However, the auction-based task allocation system relies on continuous communication to be able to correctly add new

tasks to the system and assign tasks to the correct agents during the mission execution. To cater to such communication needs, a local Wi-Fi mesh is utilized to ensure reliable communication between the agents and the base station, which also acts as the coordinator (centralized part of the auction-based task allocation) during the mission execution. Because the agents are in charge of their local autonomy and minimally interact with the auctioning system, only to send bids and receive task assignments, the entire system can be run over limited bandwidth. Data such as odometry and point clouds are required by the operators of the multi-agent system to visualize the mission execution and to meaningfully interact with the entire system by adding new tasks and follow/monitor the progress, but are not critical for the functioning of the overall architecture. The Wi-fi mesh is also used to stream such data over the network.

1) Mobile Reconfigurable Wi-Fi Mesh: During the field experiments, a mobile mesh network is deployed to provide Wi-Fi connectivity between the agents, the auctioning system, and the operator. The mesh network consists of five Wi-Fi repeaters (Fig. 13) positioned as shown in Fig. 11a. The nodes are placed so that each repeater is in the line-of-sight of the next one, to ensure a high Received Signal Strength Indicator (RSSI) for reliable and high-capacity communication between the nodes. This communication allows the task coordination to function optimally, providing more than the modest communication requirement posed by the architecture, even though the exact required bandwidth is depending both on the type of tasks and the low-level implementation, consisting of: i) announcing the available tasks to the the agents, ii) transferring the estimated costs back to the coordinator and iii) announcing the allocated tasks to the correct agents.

VI. Field Deployment (Experiments/Evaluation)

This section describes the field environment and results from a field mission, where an aerial multi-agent system was deployed to demonstrate the capabilities of the proposed task allocation and task execution system in the field. The field deployment was executed with our mining industry collaborators at Luossavaara-Kiirunavaara AB (LKAB), through the Sustainable Underground Mining (SUM) Academy programme.

A. Environment/Mission Description

The proposed multi-agent architecture is evaluated in a mission where a fleet of aerial agents is deployed (detailed in section V-A), in the challenging conditions of an underground mine to perform routine inspection tasks, with the inspection locations being set by a mine operator. Such inspection tasks are vital for identifying potential hazards and maintaining overall safety standards. During the experimental field deployment demonstration, all inspection tasks consisted of traveling to a specific location and performing a visual inspection. The mission is meticulously organized into three key phases: preparation, deployment and inspection, described in section V-B. The main mission included three aerial agents performing inspection tasks in a collaborative fashion. An operator initiated the mission by specifying the inspection locations, and the operator had the authority to add new tasks during the execution, through the interface shown in Fig. 12. Later during the mission execution, the operator used the same interface to introduce additional inspection locations into the system. The environment where the multi-agent system was deployed is illustrated in Fig. 14 and comprises of a typical subterranean mining environment with interconnected tunnels. The environment was chosen by the mining operators to depict a realistic location where such a system would be deployed to perform actual missions, such as routine inspection and gas measurement after blasting. The size was partially limited to show a working proof-of-concept, while guaranteeing the safety of the personnel and minimizing the disturbances to the daily operations.

B. Validation through large-scale Simulations

To verify the functionality of the system before deploying it in the intended mining environment, extensive validation of the architecture is first conducted in simulations. The system is simulated in a tunnel environment, similar enough to match the main challenges that would be encountered during the actual system deployment. The main reason for conducting the simulation study is to evaluate the local autonomy and task allocation functionalities in a larger environment, to assess the performance and scalability, while showcasing an example of how the architecture could be utilized.

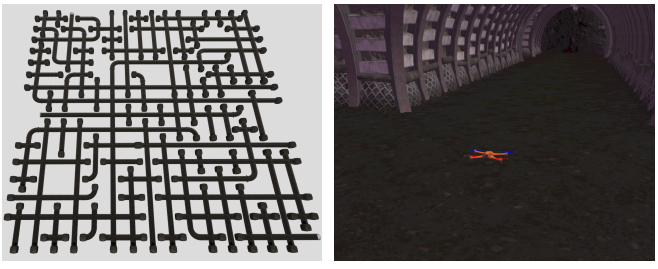
The simulation environment, shown in Fig. 15, is set-up using Gazebo Harmonic[2]. The evaluation is done using five Quadrotor agents, performing 25 routine inspection tasks, consisting of moving to a location and staying there for a fixed amount of time. The tasks are executed according to the behavior trees shown in Fig. 7, 8 and 9. To evaluate the behavior of the system, three tests are constructed in the following scenarios: i) and ii) are designed to show the execution of the routine inspections, with the agents starting at different initial locations, while iii) shows the response of the system in the case where one of the agent fails during mission execution. All scenarios consider the same environment and the same 25 inspection tasks. The resulting traversed paths during these evaluation scenarios are shown in Fig. 16 and Fig. 17 and the key metrics are summarized in Table. II. Even though the communication between the agents and the auctioning system is not explicitly modeled in the simulator, the system is evaluated for the case of agent failure by inducing an artificial failure of one agent during the mission execution. The results from the three simulated evaluation scenarios and the response of the reallocation mechanism in the case of an agent failure are included in the supplementary video that accompanies this article.

To set a baseline for the evaluation of our multi-quadrotor coordination solution for routine inspection in a mine, we also present results from a solution that uses a single quadrotor for inspection. Given a set of inspection tasks distributed

[2]https://gazebosim.org

(a) (b) (c) (d)

Fig. 14: Pictures showing the mining environment where the system was deployed and evaluated. (a) shows one of the aerial agents used to perform the inspection tasks prior to takeoff, (b) – (d) shows three snapshots of the tunnels in the underground mine where the evaluation mission was performed.



(a) (b)

Fig. 15: Snapshots illustrating the simulation environment where the architecture was tested before being deployed in the real-life mining environment. (a) shows the environment consisting of tunnels, chosen to match the challenges encountered in an underground mine, and (b) depicts the quadrotor model used during the simulation.

in an environment, we solved a traveling salesman problem (TSP) to derive the optimal sequence of locations to visit. We find that one UAV is required to travel a distance of 1535 m (visiting all inspection points, starting from the center of the simulated environment) with a total mission duration of approximately 900 seconds. In comparison, the five-drone solution presented in our work required a cumulative travel distance, averaged over the three illustrative scenarios, of 1940 m. With an average distance per UAV of 388 m and an average total mission duration spanning 503 seconds.

This demonstrates that a multi-UAV solution provides a faster solution in a large scale mine, while incurring lower overall agent travel costs in accomplishing an inspection mission.

C. Field Deployment Results

For the experimental validation of the proposed multi-agent architecture, a total of seven inspection tasks were chosen in the LKAB mine. Three Quadrotors were deployed to execute the inspection mission. The location of the seven inspection tasks $\{T_1, \ldots, T_7\}$ and the agents $\{R_1, R_2, R_3\}$ are indicated in Fig. 14. The paths tracked by the Quadrotors in the experiment are plotted in Figure 18 within the 3D point

TABLE II

TABLE SUMMARIZING THE RESULTS FROM THE THREE SIMULATION SCENARIOS WITH KEY METRICS, INCLUDING THE TOTAL DISTANCE TRAVELED BY ALL AGENTS TO COMPLETE ALL INSPECTION TASKS AND THEN RETURN TO THEIR RESPECTIVE LANDING LOCATIONS AND THE DISTANCE TRAVELED BY THE INDIVIDUAL AGENTS.

Scenario 1:

Parameter	Value
Total distance covered	1659 m
Number of agents	5
Number of inspection tasks	25
Distance traveled, agent 1	320 m
Distance traveled, agent 2	278 m
Distance traveled, agent 3	393 m
Distance traveled, agent 4	295 m
Distance traveled, agent 5	373 m

Scenario 2:

Parameter	Value
Total distance covered	2443 m
Number of agents	5
Number of inspection tasks	25
Distance traveled, agent 1	455 m
Distance traveled, agent 2	448 m
Distance traveled, agent 3	624 m
Distance traveled, agent 4	506 m
Distance traveled, agent 5	410 m

Scenario 3:

Parameter	Value
Total distance covered	1720 m
Number of agents	5
Number of inspection tasks	25
Distance traveled, agent 1	323 m
Distance traveled, agent 2	297 m
Distance traveled, agent 3	133 m
Distance traveled, agent 4	407 m
Distance traveled, agent 5	560 m

(a) (b)

Fig. 16: The paths traversed by the agents in the simulation scenarios i) and ii) are shown over a 2D projection of the environment, (a) shows the resulting traversed paths from scenario i) and (b) shows the traversed paths from scenario ii) where different initial positions are used for the agents.

TABLE III

TABLE SUMMARIZING IMPORTANT STATISTICS FROM THE EXPERIMENT. TOTAL DISTANCE TRAVELED BY ALL AGENTS TO COMPLETE ALL INSPECTION TASKS AND THEN RETURN TO THEIR RESPECTIVE LANDING LOCATIONS, MISSION DURATION, NUMBER OF LOCATIONS INSPECTED.

Parameter	Value
Mission duration	311 s
Distance to complete tasks	143.7 m
Total distance covered	294 m
Number of agents	3
Number of inspection points	7

TABLE IV

THE TABLE PRESENTS THE EXECUTION DETAILS OF THE SEVEN INSPECTION TASKS ACCOMPLISHED BY THE QUADROTORS IN THE EXPERIMENTAL DEMONSTRATION, INCLUDING THE TIME EACH TASK WAS ADDED TO THE POOL OF AVAILABLE TASKS, THE COMPLETION TIME, THE DURATION FROM AGENT ALLOCATION TO TASK COMPLETION, AND THE SPECIFIC AGENT ASSIGNED TO EACH TASK.

Task	Added [s]	Finished [s]	Execution Time [s]	Agent
T_1	0	69	69	R_1
T_2	0	46	46	R_1
T_3	0	68	68	R_2
T_4	0	115	115	R_2
T_5	0	50	50	R_3
T_6	0	103	103	R_3
T_7	116	157	41	R_2

cloud of the mine. The key parameters of the mission and the results from the experiment, including mission duration and distance traveled by the Quadrotors, are summarized in Table III. The times when tasks were added by the operator, the times when the tasks were completed, and the execution information of all the inspection tasks are detailed in Table IV. The aerial agents autonomously navigated a complex environment using on-board sensing and computation, while all the communication related to task coordination was enabled through the deployed Wi-Fi mesh.

The simulation and experimental results support the main motivation for considering a multi-quadrotor system over a single-quadrotor system for inspection missions in a mine. If only one agent is used, it would take approximately 15 minutes for the agent to perform all the inspection tasks and return to the home location, which would stress the limited battery life of the Quadrotors. Using the proposed multi-agent architecture, three Quadrotors accomplished the inspection mission in 5 minutes and 11 seconds, while traveling a total distance of 294m. During the experiment, the auctioning framework was executed at 0.2 Hz and required a bandwidth of a few hundred bytes per second. However, this bandwidth depends on the implementation details and could be optimized for an even lower data rate. The field deployment of the multi-agent system in the underground mine successfully demonstrates the performance and usability of the proposed multi-agent coordination framework.

A video recording of the experimental demonstration accompanies this article as a supplementary video and can also be viewed at: link to the video [3]. The video shows the harsh mining environment and the performance of the aerial agents during the execution of the inspection tasks. In Fig. 18, an overview of the entire mission is presented from the operator's viewpoint, along with the agents' initial locations, the location of the inspection tasks, and the paths traversed by the agents to reach the target inspection points. The experimental demonstration highlighted the proposed framework's flexibility and ease of use, confirming its potential for enhancing routine inspections in large-scale mining environments.

D. Lessons Learned

The deployment of a multi-agent system in a large-scale subterranean environment for inspection tasks has yielded multiple valuable lessons, both practical (such as dealing with perpetual hardware issues) and abstract (such as designing the overarching software structure). Firstly, the importance of reliable communication became evident, as data loss can significantly impact coordination between drones.

[3]https://youtu.be/4eyRCCRAEYg

(a) (b) (c)

Fig. 17: The paths traversed by the agents in scenario iii) of the simulation-based evaluation are shown over a 2d projection of the environment. (a) shows the initial configuration of the agents and the location of the 25 inspection locations as red cubes, (b) highlights the time instant where one agent (in the top-right corner, color coded as yellow) fails during its task execution. (c) shows the final traversed paths after all tasks are executed and the remaining agents have returned to their starting locations.

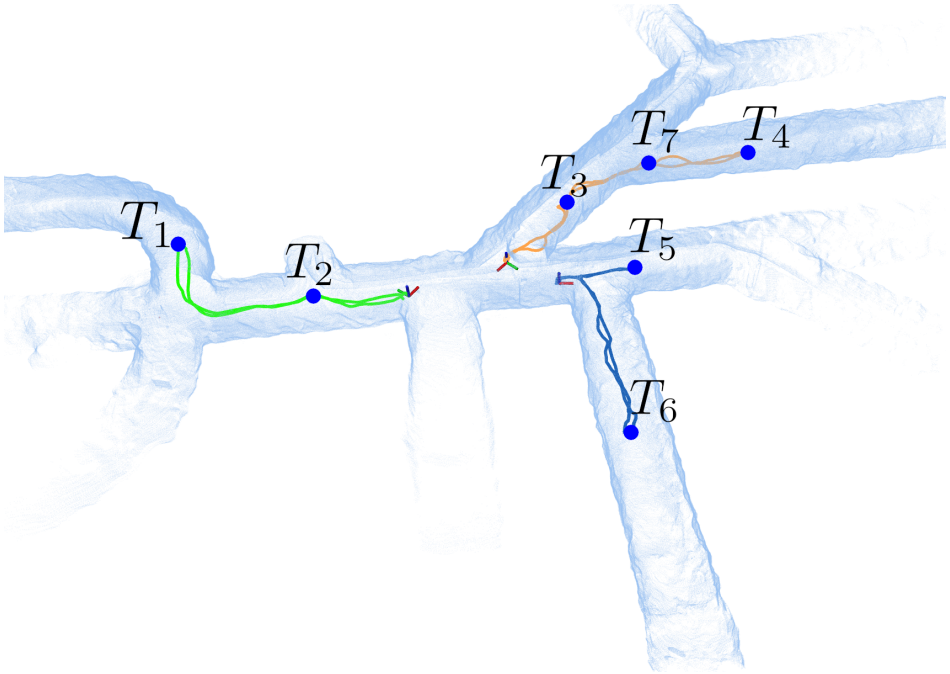


Fig. 18: The results from the experimental multi-agent inspection mission in a mine, highlighting the inspection locations and the paths traversed by the agents. The figure presents the operator's viewpoint, with the autonomous agents localized in a common frame of reference. In total, seven discrete inspection tasks, indicated in blue, were completed by the three aerial agents during the mission.

Secondly, the importance of having robust and time-tested advanced navigation, localization, and path planning modules was realized. This work, building on previous efforts in path planning and navigation, allowed for quick integration into the multi-agent architecture and deployment, due to the high level of stability and robustness already achieved by the individual modules.

1) Communication Uncertainty: The architecture is designed to utilize a baseline of mobile Wi-Fi mesh-based communication to ensure interaction between the auctioneer and the agents, so that optimal tasks are allocated at every time step. However, in edge cases where communication maybe lost, such as areas close to the edge of the communication range, the system would still be able to execute the allocated tasks since the agents' task execution is independent of communication; the agents would be able to complete the currently assigned tasks and then return to communication range to be assigned to a new task. This could nevertheless reduce the efficiency of task allocation, since suboptimal assignments could be the consequence if some agents are executing tasks outside communication range. Another consideration in executing tasks without continuous communication is the problem of guaranteeing that every task is completed, or that no task is executed twice. In the case of an agent breakdown or failure outside the communication range, the central allocator can resort to two options: i) assume that the agent completes the task, or ii) allow other agents to be allocated to the task. The two options could be mixed, for example, the central coordinator allows an allocated agent to execute a task for a certain time without any feedback, before assigning the task to a new agent. In our implementation, the

first option was chosen for the field deployment, wherein a short-duration loss of communication between an agent and the central coordinator would result in the agent completing the task and then returning to communication range to be assigned to a new task. The third scenario in Section VI-B on simulation-based validation illustrates the response of the architecture when one agent fails or loses communication during task execution, and the reallocation option is active.

2) Local Autonomy and Task Allocation: From the perspective of local agent autonomy for aerial agents, listing the agents' available actions (along with pre- and post-conditions) and then automating the synthesis of a behavior tree for task execution proved efficient, and the use of behavior trees provided a standardized way to monitor individual agents during task execution. This approach allows for easier restructuring or adaptation of a mission to incorporate both new types of tasks and robotic platforms. This approach also allows the inclusion of tasks that consist of multiple stages, pick-and-place tasks being an example (explored in [37], where the stages need to be executed in a sequence. These aspects illustrate the advantages of flexibility and modularity offered by behavior trees in designing customizable, mission-specific local autonomy, while being robust during the design and execution phases to errors, since all agent capabilities can be individually tested and verified before building a behavior tree. In the overall multi-agent coordination architecture, the choice of a centralized task coordinator and independent task execution by the agents was made, with minimal communication exchange between the agents and the coordinator. This architecture efficiently utilized the minimal communication infrastructure setup using the WiFi mesh.

An important goal of this work was to handle the dynamic scenario where an operator could add new tasks into the system at any point. A centralized task allocation approach proved beneficial, as it provided a convenient way for the operator to interact with the overall system and register a new task into the system. However, some limitations were encountered, especially due to reactive task planning over a very short horizon. Occasionally, inefficient/suboptimal solutions emerged over time, highlighting the need for a real-time and computationally efficient way to incorporate either task clustering or planning into the task coordination architecture to address this issue. To tackle the issue of scaling to very large scenarios, where a large group of agents are spread in a large subterranean environment, local bidding strategies, as explored in [44], could be employed, which necessitates agents to estimate costs for only tasks in a local neighborhood of the agent and thereby increasing both the scalability and the reactivity of the solution. In addition, if the reactive solution, optimal over a one-step horizon, is deemed insufficient, the auction system can be expanded to allow agents to bid for bundles of tasks instead of individual tasks; this would provide allocations that are optimal over a longer horizon, while sacrificing some reactivity due to the extra computation time required to estimate the cost for executing a bundle of tasks.

3) From Design to Deployment: The selection of initial locations of the agents in the mine is a crucial part of deploying the proposed aerial multi-agent system, as it directly influences the task allocation and overall system performance. Any available knowledge of the distribution of inspection tasks should be considered in the initial positioning of agents to optimize the task execution. This is evident from the results in Table II and comparing the total distance traveled during scenarios i) and ii). Even though both scenarios involve the execution of the same 25 inspection tasks, the total distance traveled during scenario i) is 1659 m while the total distance traveled during scenario ii) is 2443 m. This reflects the need for carefully considering the initial placement of agents when the proposed multi-agent architecture is to be deployed in a mine for performing routine inspections efficiently.

The development, verification and performance evaluation of the proposed multi-agent coordination architecture progressed through multiple stages, involving evaluation of the system in simulated environments with simulated agents, followed by small-scale field testing in a subterranean environment to assess the real-life functionality of all modules and finally, the system as a whole was deployed in an operating underground mine for the final proof-of-concept demonstration. The simulation phase was crucial in the refinement of the local autonomy/task execution mechanisms and the validation of the central task allocation before physical deployment. As an instance, the system's response to the scenario where certain agents lose communication in field deployments in a subterranean environment can be evaluated thoroughly first in simulations. The second phase, with small-scale deployment using real robot platforms, allowed the verification of the functionality of additional modules, such as localization, low-level control for path tracking, and collision avoidance, that are required for the final deployment. In this work, we have pushed the TRL levels of our technology to 5-6, by demonstrating a successful proof-of-concept field deployment in a relevant operational subterranean environment. This was progressively achieved by upgrading previously demonstrated small-scale lab evaluation with Turtlebot3 robots [28] to a successful field deployment, through improvements in the core auctioning system and behavior trees and through integration with additional autonomy modules necessary for field deployments.

VII. Conclusion

The proposed multi-agent framework for deploying aerial agents in an underground mine for routine inspection missions represents a significant advancement of the state-of-the-art in autonomous inspection and industrial automation. The proposed framework allows for efficient, precise, and reliable inspection of complex and dangerous environments, greatly reducing the risk for human operators and enhancing the overall safety of mining operations. Several key capabilities, such as reactive and flexible task allocation and robust execution of autonomous aerial inspection tasks, have been demonstrated through a large-scale mission in an

operational underground mine. Furthermore, the framework demonstrated the flexibility to accommodate new inspection tasks by a mining operator, both before and during execution, through an easy-to-use user interface. This ensures that the system meets the needs of the mining operators and can be used in real large-scale mining environments.

In summary, the proposed autonomous multi-agent framework enhances current inspection capabilities in subterranean environments and also sets the stage for broader applications in industrial automation, due to the capability of the framework to generalize to both new types of tasks and additional robotic platforms.

VIII. Acknowledgments

The authors would like to acknowledge and thank their mining industry collaborators at Luossavaara-Kiirunavaara AB (LKAB). The field demonstrations included in this manuscript were enabled through the Sustainable Underground Mining (SUM) Academy programme, which is funded by LKAB and the Swedish Energy Agency, and were executed at a depth of over 500m in an active production area of the mine located in Kiruna, Sweden.

References

[1] S. Ge, F.-Y. Wang, J. Yang, Z. Ding, X. Wang, Y. Li, S. Teng, Z. Liu, Y. Ai, and L. Chen, "Making standards for smart mining operations: Intelligent vehicles for autonomous mining transportation," *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 3, pp. 413–416, 2022.

[2] N. Stathoulopoulos, B. Lindqvist, A. Koval, A.-A. Agha-Mohammadi, and G. Nikolakopoulos, "FRAME: A Modular Framework for Autonomous Map Merging: Advancements in the Field," *IEEE Transactions on Field Robotics*, vol. 1, pp. 1–26, 2024.

[3] N. Stathoulopoulos, A. Koval, and G. Nikolakopoulos, "Irregular Change Detection in Sparse Bi-Temporal Point Clouds Using Learned Place Recognition Descriptors and Point-to-Voxel Comparison," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 8077–8083.

[4] K. Ebadi, L. Bernreiter, H. Biggie, G. Catt, Y. Chang, A. Chatterjee, C. E. Denniston, S.-P. Deschênes, K. Harlow, S. Khattak, L. Nogueira, M. Palieri, P. Petráček, M. Petrlík, A. Reinke, V. Krátký, S. Zhao, A.-a. Agha-mohammadi, K. Alexis, C. Heckman, K. Khosoussi, N. Kottege, B. Morrell, M. Hutter, F. Pauling, F. Pomerleau, M. Saska, S. Scherer, R. Siegwart, J. L. Williams, and L. Carlone, "Present and Future of SLAM in Extreme Environments: The DARPA SubT Challenge," *IEEE Transactions on Robotics*, vol. 40, pp. 936–959, 2024.

[5] J. Shahmoradi, E. Talebi, P. Roghanchi, and M. Hassanalian, "A comprehensive review of applications of drone technology in the mining industry," *Drones*, vol. 4, no. 3, p. 34, 2020.

[6] L. Dunnington and M. Nakagawa, "Fast and safe gas detection from underground coal fire by drone fly over," *Environmental Pollution*, vol. 229, pp. 139–145, 2017.

[7] R. Turner, N. Bhagwat, L. Galayda, C. Knoll, E. Russell, and M. MacLaughlin, "Geotechnical characterization of underground mine excavations from uav-captured photogrammetric & thermal imagery," in *52nd US Rock Mechanics/Geomechanics Symposium*. OnePetro, 2018.

[8] T. Bamford, K. Esmaeili, and A. P. Schoellig, "Aerial rock fragmentation analysis in low-light condition using uav technology," *arXiv preprint arXiv:1708.06343*, 2017.

[9] G. Freire and R. Cota, "Capture of images in inaccessible areas in an underground mine using an unmanned aerial vehicle," in *UMT 2017: Proceedings of the First International Conference on Underground Mining Technology*. Australian Centre for Geomechanics, 2017.

[10] H. Kim and Y. Choi, "Development of a ROS-based autonomous driving robot for underground mines and its waypoint navigation experiments," *Tunnel and Underground Space*, vol. 32, no. 3, pp. 231–242, 2022.

[11] E. Protopapadakis, C. Stentoumis, N. Doulamis, A. Doulamis, K. Loupos, K. Makantasis, G. Kopsiaftis, and A. Amditis, "Autonomous robotic inspection in tunnels," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 3, pp. 167–174, 2016.

[12] DARPA. Subterranean challenge (SubT). [Online]. Available: https://www.subtchallenge.com/

[13] M. Tranzatto, T. Miki, M. Dharmadhikari, L. Bernreiter, M. Kulkarni, F. Mascarich, O. Andersson, S. Khattak, M. Hutter, R. Siegwart, and K. Alexis, "Cerberus in the darpa subterranean challenge," *Science Robotics*, vol. 7, no. 66, p. eabp9742, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abp9742

[14] A. Reinke, M. Palieri, B. Morrell, Y. Chang, K. Ebadi, L. Carlone, and A.-A. Agha-Mohammadi, "LOCUS 2.0: Robust and Computationally Efficient Lidar Odometry for Real-Time 3D Mapping," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9043–9050, 2022.

[15] K. Chen, B. T. Lopez, A.-a. Agha-mohammadi, and A. Mehta, "Direct lidar odometry: Fast localization with dense point clouds," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2000–2007, 2022.

[16] S. Khattak, H. Nguyen, F. Mascarich, T. Dang, and K. Alexis, "Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2020, pp. 1024–1029.

[17] T. Dang, F. Mascarich, S. Khattak, C. Papachristos, and K. Alexis, "Graph-based path planning for autonomous robotic exploration in subterranean environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3105–3112.

[18] J. Bayer, P. Cızek, and J. Faigl, "Autonomous multi-robot exploration with ground vehicles in darpa subterranean challenge finals," 2023.

[19] B. Lindqvist, S. Karlsson, A. Koval, I. Tevetzidis, J. Haluška, C. Kanellakis, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "Multimodality robotic systems: Integrated combined legged-aerial mobility for subterranean search-and-rescue," *Robotics and Autonomous Systems*, vol. 154, p. 104134, 2022.

[20] T. Rouček, M. Pecka, P. Čížek, T. Petříček, J. Bayer, V. Šalanskỳ, D. Heřt, M. Petrlík, T. Báča, V. Spurnỳ, *et al.*, "Darpa subterranean challenge: Multi-robotic exploration of underground environments," in *Modelling and Simulation for Autonomous Systems: 6th International Conference, MESAS 2019, Palermo, Italy, October 29–31, 2019, Revised Selected Papers 6*. Springer, 2020, pp. 274–290.

[21] Y. Chang, K. Ebadi, C. E. Denniston, M. F. Ginting, A. Rosinol, A. Reinke, M. Palieri, J. Shi, A. Chatterjee, B. Morrell, *et al.*, "Lamp 2.0: A robust multi-robot slam system for operation in challenging large-scale underground environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9175–9182, 2022.

[22] M. O'Brien, J. Williams, S. Chen, A. Pitt, R. Arkin, and N. Kottege, "Dynamic task allocation approaches for coordinated exploration of subterranean environments," *Autonomous Robots*, vol. 47, no. 8, pp. 1559–1577, 2023.

[23] B. Lindqvist, C. Kanellakis, S. S. Mansouri, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "Compra: A compact reactive autonomy framework for subterranean mav based search-and-rescue operations," *Journal of Intelligent & Robotic Systems*, vol. 105, no. 49, 2022. [Online]. Available: https://doi.org/10.1007/s10846-022-01665-6

[24] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems," *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–31, apr 2019.

[25] M. B. Dias, "Traderbots: A new paradigm for robust and efficient multirobot coordination in dynamic environments," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, January 2004.

[26] G. P. Das, T. M. McGinnity, S. A. Coleman, and L. Behera, "A distributed task allocation algorithm for a multi-robot system in healthcare facilities," *Journal of Intelligent & Robotic Systems*, vol. 80, no. 1, pp. 33–58, Oct 2015. [Online]. Available: https://doi.org/10.1007/s10846-014-0154-2

[27] S. Sariel-Talay, T. R. Balch, and N. Erdogan, "A generic framework for distributed multirobot cooperation," *Journal of Intelligent & Robotic Systems*, vol. 63, no. 2, pp. 323–358, Aug 2011. [Online]. Available: https://doi.org/10.1007/s10846-011-9558-4

[28] N. Dahlquist, B. Lindqvist, A. Saradagi, and G. Nikolakopoulos, "Reactive multi-agent coordination using auction-based task allocation and behavior trees," in *2023 IEEE Conference on Control Technology and Applications (CCTA)*, 2023, pp. 829–834.

[29] D. Hert, T. Baca, P. Petracek, K. Vit, R. Penicka, V. Spurny, M. Petrlik, M. Vrba, D. Zaitlik, P. Stoudek, V. Walter, P. Stepan, J. Horyna, V. Pritzl, M. Sramek, A. Ahmad, G. Silano, D. B. Licea, P. Stibinger, T. Nascimento, and M. Saska, "Mrs drone: A modular platform for real-world deployment of aerial multi-robot systems," *Journal of Intelligent & Robotic Systems*, vol. 108, no. 64, 2023. [Online]. Available: https://doi.org/10.1007/s10846-023-01879-2

[30] A. G. Araújo, C. A. P. Pizzino, M. S. Couceiro, and R. P. Rocha, "A multi-drone system proof of concept for forestry applications," *Drones*, vol. 9, no. 2, 2025. [Online]. Available: https://www.mdpi.com/2504-446X/9/2/80

[31] S. Amiri, K. Chandan, and S. Zhang, "Reasoning with scene graphs for robot planning under partial observability," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5560–5567, 2022.

[32] L. E. Holloway, B. H. Krogh, and A. Giua, "A survey of petri net methods for controlled discrete event systems," *Discrete Event Dynamic Systems*, vol. 7, no. 2, pp. 151–190, Apr 1997. [Online]. Available: https://doi.org/10.1023/A:1008271916548

[33] M. Ghallab, D. Nau, and P. Traverso, "The Actor's View of Automated Planning and Acting: A Position Paper," *Artificial Intelligence*, vol. 208, pp. 1 – 17, Mar. 2014. [Online]. Available: https://hal.science/hal-01982043

[34] M. Colledanchise and P. Ögren, "Behavior trees in robotics and ai," Jul 2018.

[35] M. Iovino, J. Förster, P. Falco, J. J. Chung, R. Siegwart, and C. Smith, "On the programming effort required to generate behavior trees and finite state machines for robotic applications," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2023.

[36] M. Colledanchise, D. Almeida, and P. Ögren, "Towards blended reactive planning and acting using behavior trees," in *2019 International Conference on Robotics and Automation (ICRA) Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019*. IEEE, 2019.

[37] N. Dahlquist, A. Saradagi, and G. Nikolakopoulos, "Behavior tree based decentralized multi-agent coordination for balanced servicing of time varying task queues," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 5718–5723.

[38] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.

[39] N. Stathoulopoulos, A. Koval, and G. Nikolakopoulos, "3DEG: Data-Driven Descriptor Extraction for Global re-localization in subterranean environments," *Expert Systems with Applications*, vol. 237, p. 121508, 2024.

[40] S. Karlsson, A. Koval, C. Kanellakis, and G. Nikolakopoulos, "D+*: A risk aware platform agnostic heterogeneous path planner," *Expert systems with applications*, vol. 215, p. 119408, 2023.

[41] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, no. 3, pp. 189–206, 2013.

[42] B. Lindqvist, J. Haluska, C. Kanellakis, and G. Nikolakopoulos, "An adaptive 3d artificial potential field for fail-safe uav navigation," in *2022 30th Mediterranean Conference on Control and Automation (MED)*. IEEE, 2022, pp. 362–367.

[43] A. Patel, B. Lindqvist, C. Kanellakis, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "REF: A Rapid Exploration Framework for Deploying Autonomous MAVs in Unknown Environments," *arXiv preprint arXiv:2205.15670*, 2022.

[44] N. Dahlquist, A. Saradagi, and G. Nikolakopoulos, "Local bidding strategies for reactive and scalable auction-based multi-agent coordination," in *2024 32nd Mediterranean Conference on Control and Automation (MED)*, 2024, pp. 203–208.