2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

HMAAC: Hierarchical Multi-Agent Actor-Critic for Aerial Search with Explicit Coordination Modeling

Chuanneng Sun, Songjun Huang, and Dario Pompili

***Abstract*— Unmanned Aerial Vehicles (UAVs) have become prevalent in Search-And-Rescue (SAR) missions. However, existing solutions to the control and coordination of UAVs are mostly limited to specific environments and are not robust to handle unreliable/unstable communications. To deal with these challenges, Hierarchical Multi-Agent Actor-Critic (HMAAC) framework is proposed where a high-level policy is placed on top of individual low-level actor-critic policies to relax the inter-dependency among the agents. The low-level policies are considered conditionally independent given the coordination action, which is generated by the high-level policy. A Centralized Training Decentralized Execution (CTDE) would not work because it cannot be assumed that communication is always perfect during training and that the whole system can rely on stable communications during deployment. The proposed framework is evaluated in AirSim, a realistic multi-UAV simulator, and is compared against two existing algorithms, i.e., Multi-Agent Actor-Critic (MAAC) and decentralized REINFORCE, in two scenarios, (a) when packet drop is modeled as a Bernoulli process and (b) when shadow zones are created in the search space and communication will be lost if the agents are in these zones. Results show that HMAAC is scalable and robust to unreliable communication and outperforms the other algorithms in terms of exploration and coordination when the number of agents is large and communications are not stable.**

I. Introduction

Overview: In a disaster response, Search-And-Rescue (SAR) is a crucial part because human lives depend on it. However, it is impractical and risky to deploy rescue personnel directly into the field because the environment is unknown and generally dynamic [1]–[3]. A promising and achievable solution would be to deploy unmanned robots to perform multi-robot SAR operations instead of relying on human beings. Remotely-controlled robots operated by professionals can guarantee good performance, but such an approach is not scalable since it requires much manpower when the number of robots gets large. Therefore, intelligent robots such as Unmanned Aerial Vehicles (UAVs) have become a good choice for their low cost, scalability, and flexibility (see Fig. 1). We focus on the search part of multi-robot SAR [4], [5], which can be further extended to the search for more generalized targets such as specific persons, vehicles, or valuable equipment.

Motivation: The main challenges in multi-agent SAR missions are the *communication and coordination* of the multiple UAVs. On the one hand, generally, multiple agents can accelerate the process to reach the goal of the mission; on the other hand, a bad coordination and communication strategy could lead to outcomes that are even worse than single-agent deployment (e.g., collisions between agents). Furthermore, from a single agent's point of view, the environment is non-stationary (i.e., *partially observable*) because of the actions the other agents take, which makes the environment unpredictable for any individual agent. If each agent were able to observe the states and actions of the other agents, the multi-agent problem would boil down to a single-agent one; the requirement for such *full observability*, however, can hardly be achieved in real-world SAR missions, and the only way to approach it is through communication. In reality, however, the communication condition is not stable and reliable, especially in disaster scenes such as earthquakes and floods where APs could be damaged or oversubscribed. In such scenarios, communication among the UAVs cannot be guaranteed and it is possible that, at any time during the mission, any agent could become isolated from the others. Thus, how to act when a subset of agents is isolated is a great challenge to the coordination of multiple agents under unstable communications. Although communication issues have been discussed in multi-agent machine learning, such as Multi-Agent Reinforcement Learning (MARL) [6], [7] and federated learning [8], [9], most of the works do not consider disconnection issues but focus on reducing bandwidth requirements. Many existing works on SAR use model-based methods [10], [11] and only focus on a single-robot scenario [12], [13]. Those that involve multiple robots either do not consider coordination aspects among agents [14] or do not take communication problems into account [15], [16].

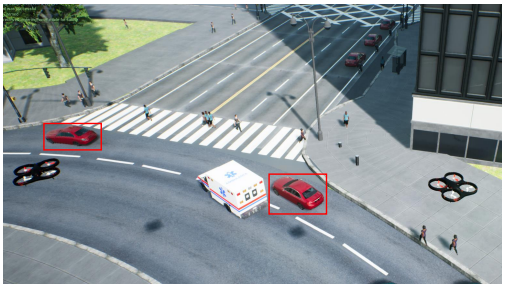


Fig. 1: Two Unmanned Aerial Vehicles (UAVs) are executing target search missions to locate red vehicles.

Our Approach: To tackle the aforementioned challenges, we propose a distributed Hierarchical Multi-Agent Actor-Critic (HMAAC) algorithm for aerial SAR missions. Consider a disaster scene with obstacles and victims scattered in the area. The goal of the framework is to control the UAVs in such a way as to explore the entire unknown environment

This work is supported by the NSF RTML Award No. CCF-1937403. The authors are with the Dept. of Electrical and Computer Engineering, Rutgers University–New Brunswick, NJ, USA. Emails: {*chuanneng.sun, songjun.huang, pompili*}*@rutgers.edu*

(i.e., with zero prior knowledge about the environment and the targets) while avoiding (i) overlaps between the UAVs' observations and (ii) collisions with surrounding obstacles and other UAVs.

Thus, in the proposed HMAAC framework, we introduce a high-level policy on top of each agent, which can generate coordination commands. We assume that the low-level agents are conditionally independent of each other given the coordination action generated by the high-level policy, and can make decisions and updates locally even when the communication link is temporarily broken. We assume that the UAVs connect to a device that can monitor the UAVs' status in real time. This device is operated by a rescue operator and it can be regarded as a network Coordination Point (CP) since every UAV needs to communicate with it. Unlike most MARL frameworks [5], [7], [17], we do *not* adopt the Centralized Training Decentralized Execution (CTDE) pipeline, because (i) sometimes we do not have perfect communication during training and (ii) multi-agent systems can benefit from communication among agents and a fully decentralized deployment would lose this benefit.

The proposed HMAAC framework is robust to unreliable communication conditions, even during training. Instead of talking to each other at each time step, each agent only needs to talk to the CP to obtain the coordination action. During disconnection for an individual agent, it will use the coordination action from the last successful communication. Besides that, our framework can also work for the multi-team scenario where multiple robots form multiple teams and these robots can have different types and configurations or even come from different parties (e.g., a joint SAR mission conducted by multiple countries, each with its own robots). In such scenarios, a centralized communication structure can greatly help the coordination between the teams/robots toward better performance. Although a centralized structure might suffer from single-point failure (i.e., the CP suddenly breaks down), we argue that in a SAR mission, there will be sufficient resources on the ground to be used as a backup (e.g., a backup vehicle).

Our **contributions** can be summarized as follows:

- We propose HMAAC, a hierarchical MARL framework that introduces another hierarchy on top of the multi-agent actor-critic framework by adding a high-level policy that observes the states of all low-level policies to achieve better coordination during disconnection.
- We relax the inter-dependency among the agents and allow the actors and critics to perform updates with their own states and actions even during disconnection.
- We evaluate HMAAC in an aerial SAR mission in AirSim [18] under two conditions, (i) unreliable communications, where communication fails with a probability and (ii) shadow zones created by buildings. HMAAC is compared with Multi-Agent Actor-Critic (MAAC) [3] and REINFORCE [19]; the results show that HMAAC performs better in terms of scalability and robustness to unreliable communication as it outperforms other frameworks when the number of agents gets large and the communication condition unreliable.

II. Related Work

We position now our work in the context of others' work. We first discuss existing SAR frameworks, including feature-based and learning-based methods. Then, we review MARL methods with communication aspects.

Aerial SAR: Existing aerial SAR frameworks can be categorized into two types–optimization based and Reinforcement Learning (RL) based. Delmerico et al. [20] proposed an active search framework in which drones scan and classify the terrain to find feasible paths for ground robots to reach the targets. The objective function in this problem is defined as the time for the ground robots to cover the waypoints proposed by drones. In terms of learning-based frameworks, Sadhu et al. [3] proposed a distributed MAAC framework for SAR missions, where each agent is deployed with a pair of actor and critic. The critics take as input the agent's state and the actions of all agents to calculate the action value. This framework can handle the disconnection issues but, as the number of UAVs increase, the requirement for communication bandwidth grows significantly. Sampedro et al. [21] designed a fully-autonomous UAV equipped with Convolutional Neural Network (CNN) learning model and a Deep Deterministic Policy Gradients (DDPG)-based algorithm to deal with Image-Based Visual Servoing (IBVS) tasks in rescue missions with an acceptable computational cost. However, the performance is only evaluated in an indoor environment. Igoe et al. [5] proposed a RL framework using perfect communication during training and imperfect communication during testing. A latent state distribution is used to model the system state and the reward is defined based on the Maximum a Posterior (MAP) of the state distribution. For continuous actions, they used the Poisson Point Process formulation to model the generation of targets. Nonetheless, this framework assumes perfect communication during training, which is unrealistic because the offline training environment often deviates from the real environment.

MARL: In the area of MARL, much work has been conducted, but only a few works take unreliable communication into consideration. Lowe et al. [17] proposed MAAC, which extends the actor-critic framework to the multi-agent scenario. In MAAC, the critic receives the states and actions from all agents during training to guide the agents' coordination while each actor only depends on its own state. Although this framework can perform well, it only works for frameworks that have reliable communication during training. Furthermore, they assume that there is no communication during deployment/testing; it is obvious, however, that with communication, the agents can perform even better. Das et al. [22] proposed TarMAC, which is an actor-critic framework with decentralized actors and a centralized critic; targeted communication is achieved in their work by integrating the attention mechanism and, at each time step, the final message is generated by running the attention calculation process multiple rounds. Han et al. [23] proposed a hierarchical RL framework based on options,

which can be regarded as a set of sub-policy, for multi-agent systems. Each agent will select an option and execute the option until it terminates. To ensure full observability, the agent needs to broadcast the option it selects whenever the last option terminates; and, to reduce the communication overhead, an on-demand communication mechanism is designed. Foerster et al. [7] proposed two communication frameworks, namely DIAL and RIAL; the latter regards the communication message as an action and trains a Q network to structure the message, while in the former, the message coming out from one agent's NN goes into another agent's NN as input and, once the gradient is calculated, it will be transmitted back to the former agent. These approaches can perform well if the communication is stable but they do not have the robustness against unreliable communications where temporary disconnections could happen.

III. Proposed Solution

We now first introduce the background and notations for MARL in Sect. III-A and then we present our HMAAC framework in Sect. III-B. HMAAC introduces a high-level policy to model coordination explicitly and the low-level policies are assumed to be conditionally independent given the coordination action. Unlike MAAC, HMAAC does not require the states and actions of other agents to perform an update; instead, it only needs the coordination action and its own state.

A. Background and Notations

MARL can be modeled with Partially Observable Markov Decision Process (POMDP), an extension to a multi-agent manner of the Markov Decision Process (MDP). An MDP for N agents consists of a set of states $\mathcal{S}$, which describes all the configurations for the participating agents, a set of actions $\mathcal{A}_1, ..., \mathcal{A}_N$ and a set of observations $\mathcal{O}_1, ..., \mathcal{O}_N$. Each agent i possesses a policy $\boldsymbol{\pi}_i : \mathcal{O}_i \times \mathcal{A}_i \mapsto [0, 1]$ parameterized by θ_i. The environment will generate the next state based on the state transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A}_1 \times ... \times \mathcal{A}_N \mapsto \mathcal{S}$. Each agent will receive a reward from the environment as a function of state and action $r_i : \mathcal{S} \times \mathcal{A}_i \mapsto \mathbb{R}$ as well as an individual observation that is correlated with the state, $o_i : \mathcal{S} \mapsto \mathcal{O}_i$. Each agent tries to maximize its total expected return $R_i = \sum_{t=0}^{T} \gamma^t r_i^t$, where γ is a discount factor and T is the total time length.

B. Hierarchical Multi-Agent Actor-Critic

In MAAC, due to the partial observability, each agent requires the states and actions information from the others to make a decision, implying that the agents are interdependent on each other. In reality, this dependency is realized via the continuous communication among the agents. However, this requirement can hardly be guaranteed, especially in disaster scenes where CP could be damaged or oversubscribed. The main reason for this requirement is that the critic in MAAC needs global information to perform an update towards better coordination. The graphical model showing that each agent in MAAC depends on every other agent is shown in Fig. 2a.

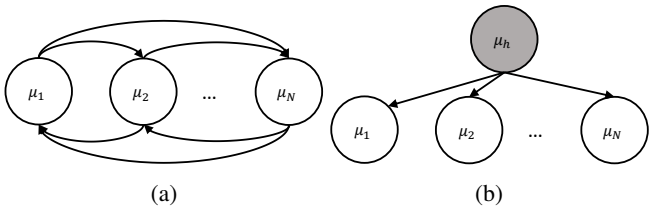


Fig. 2: Graphical models for (a) Multi-Agent Actor-Critic (MAAC) and (b) our Hierarchical MAAC (HMAAC). Adding a high-level policy as a latent variable removes the inter-dependency among the low-level policies.

To relax this inter-dependency, we add a high-level policy to the MAAC framework; the low-level policies will take local observation and the coordination action produced by the high-level policy as input to generate the local action. The high-level policy can also be regarded as a latent variable, and the low-level agents are conditionally independent given the latent variable (Fig. 2b). Consider that each agent has a deterministic policy $\boldsymbol{\mu}_i$ parameterized by θ_i, then the joint policy of all low-level agents, $\boldsymbol{\mu}_{jt}$, can be represented as $\boldsymbol{\mu}_{jt}(a_1, ..., a_N|\boldsymbol{o}, a_c) = \prod_{i=1}^{N} \boldsymbol{\mu}_i(a_i|a_{i+1}, ..., a_N, a_c, \boldsymbol{o}) = \prod_{i=1}^{N} \boldsymbol{\mu}_i(a_i|a_c, o_i)$, where $\boldsymbol{o} = \{o_1, ..., o_N\}$ are the observations from all agents and a_c is the coordination action. By querying the high-level policy to obtain the coordination action, the agents will be able to make decisions without directly talking with other agents and perform local optimizations on their own. Furthermore, the same equation holds for stochastic policies with discrete action spaces. Unlike MAAC, where the critic needs the states and actions from all agents, the local critic in HMAAC only takes the states and actions information from its corresponding actor for the prediction of the action-value function. The global information for coordination is provided by the high-level policy, which can observe the states of all agents.

To formally describe the proposed framework, let $\boldsymbol{\mu}_h$ denote the high-level policy parameterized by θ_h as a function of all agents' observations. Actor-network of agent i takes the local state, o_i, as input and outputs a deterministic action while the high-level policy $\boldsymbol{\mu}_h$ takes the observations of all agents $\boldsymbol{o}$ as input and outputs the coordination action $a_c \in \mathcal{A}_C$, where $\mathcal{A}_C$ is the coordination action space. In our framework, there are two sets of parameters that need to be optimized–the local agents' parameters $\{\theta_1, ..., \theta_N\}$ and the high-level policy's parameters θ_h. If an agent can communicate with the CP, it will then upload its local observation to the CP to get the latest coordination action a_c; if the agent cannot communicate with the CP, it will use the coordination action from the last successful communication.

Now, we will derive the policy gradient for these two sets of parameters. The overall objective of MARL is to optimize the accumulated rewards of the agents, which can be represented by $J(\theta_i, \theta_h) = \mathbb{E}[R_i]$. The gradient of this objective function w.r.t. the local policy model, $\nabla_{\theta_i} J(\theta_i, \theta_h)$,

can be written as,

$$\nabla_{\theta_i} J(\theta_i, \theta_h) = \mathbb{E}_{o,a,a_c\sim\mathcal{D}_i}[\nabla_{\theta_i} Q_i^{\boldsymbol{\mu}}(o_i, a_i)|_{a_i=\boldsymbol{\mu}_i(a_i|o_i,a_c)}] \tag{1}$$

$$= \mathbb{E}_{o,a,a_c\sim\mathcal{D}_i}[\nabla_{\theta_i}\boldsymbol{\mu}_i(a_i|o_i, a_c)\nabla_{a_i} Q_i^{\boldsymbol{\mu}}(o_i, a_i)]. \tag{2}$$

As for the high-level policy, it will be updated along with each agent. Specifically, for agent i, if it can communicate with the CP, then the gradient for the high-level policy on agent i's side, $\nabla_{\theta_h} J(\theta_i, \theta_h)$, can be written as,

$$\nabla_{\theta_h} J(\theta_i, \theta_h) = \tag{3}$$

$$= \mathbb{E}_{o,a\sim\mathcal{D}_i}[\nabla_{\theta_h} Q_i^{\boldsymbol{\mu}}(o_i, a_i)|_{a_i=\boldsymbol{\mu}_i(a_i|o_i,\boldsymbol{\mu}_h(\mathbf{o}))}] \tag{4}$$

$$= \mathbb{E}_{o,a\sim\mathcal{D}_i}[\nabla_{\theta_h}\boldsymbol{\mu}_h(a_c|\mathbf{o})\nabla_{a_c}\boldsymbol{\mu}_i(a_i|o_i, a_c)\nabla_{a_i} Q_i^{\boldsymbol{\mu}}(o_i, a_i)], \tag{5}$$

where $\mathcal{D}_i$ represents the local Experience Replay (ER) buffer for agent i, which contains tuples $< o_i, a_i, a_c, r_i, o_i' >$. Note that the ER buffer only stores the local information without the need for any global information. The critic for agent i, $Q_i^{\boldsymbol{\mu}}$, is updated by the loss function defined as,

$$\mathcal{L}(\theta_i) = \mathbb{E}_{o,a\sim\mathcal{D}_i}[(Q_i^{\boldsymbol{\mu}}(o_i, a_i) - y)^2], \tag{6}$$

$$y = r_i(o_i, a_i) + \gamma\, Q_i^{\boldsymbol{\mu}'}(o_i', a_i')\big|_{a_j'=\boldsymbol{\mu}_j'(o_j')}, \tag{7}$$

where $\boldsymbol{\mu}_j'$ is the target policy with delayed parameters θ_j' for agent j. When the communication link works, the local agents will follow (2) and (5) to update the actors and critics. One thing worth noticing is that, unlike in (5), where a_c is generated from the high-level policy, in (2) we sample the coordination actions directly from the ER memory because we do not need the gradients to backpropagate through the high-level policy during the local update.

Fig. 3 provides a diagram of HMAAC. During real-time decision-making, the high-level policy will first generate its action a_c and transmit it to the agents. Then, individual agents will make decisions based on their local observation o_i and the coordination action. After executing the actions, the agent will receive a reward and the next observation from the environment, which is represented by black solid lines in the figure. To update the models, mini-batches are sampled from the ER memory and passed to actors and critics for loss calculation, as shown with red dashed lines. If the communication between one UAV and the CP is good, then it will upload the loss for high-level policy to update. A more formal description of the framework is in Algo. 1.

IV. Performance Evaluation

We now first introduce the setup for the experiments and then evaluate the performance of our solution under two scenarios: (i) unreliable communications, where packets will drop following the Bernoulli process and, as a result, the communication between UAVs and the CP would fail following a failure probability p (Sect. IV-A); (ii) shadow zones, where there will be certain zones (e.g., the area blocked by a building) in which the UAVs will lose communication (Sect. IV-B).

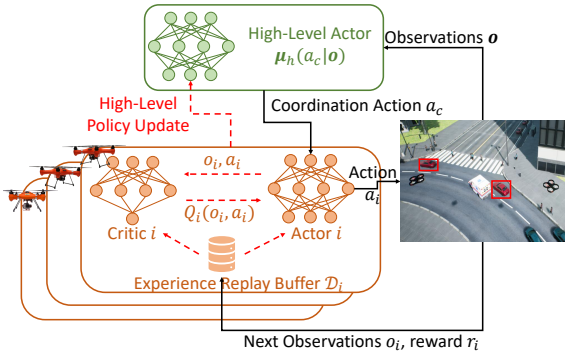


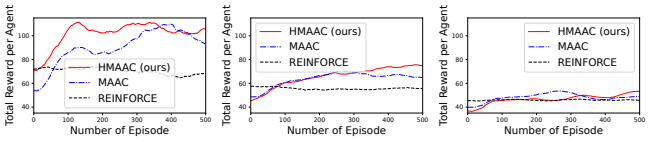
Fig. 3: Workflow of the proposed framework, where black solid lines indicate the forward inference and red dashed lines represent update/learning information flows.

Algorithm 1: HMAAC Algorithm

```
Initialize random noise 𝒩 for exploration;
Receive initial observations x;
for episode = 1, 2, ... do
    for step=1, 2, ... do
        Upload agents' observations o to the CP;
        Calculate coordination action a_c = μ_h(a_c|o);
        for agent i=1, ..., N do
            if communication is available then
                Receive a_c from CP;
            else
                Use a_c from last communication;
            Select action a_i = μ_i(a_i|o_i, a_c) + 𝒩_i;
            Execute a_i to get r_i and o_i';
            Update μ_i and Q_i via (2) and (6);
            if communication is available then
                Compute gradients for μ_h via (5);
                Upload gradients to CP to update μ_h;
```

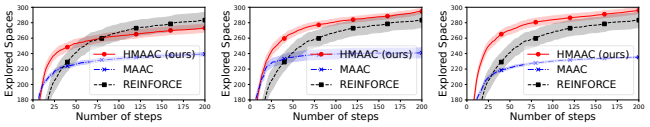
Comparison Plan: We compare the proposed HMAAC against two algorithms: (i) MAAC [3], where each UAV observes/receives the states and actions of all other UAVs to update its critic and independently updates its actors based on its own experiences; when one UAV is isolated (i.e., cannot communicate with others), it cannot receive the states and actions from others, and therefore cannot perform the update for both its actor and critic. (ii) REINFORCE [19], where each agent has a policy network and updates it only based on its own states and actions; in this scenario, no communication is needed because it is a decentralized framework.

Environment Setup and Definition: The experiments are conducted in AirSim [24], a simulator developed using Unreal Engine for realistic visual simulation. To do experiments, the environment for the problem needs to be formally defined. For a UAV, we assume that it can translate and rotate freely. The *state* of agent i consists of its position, angle, and an ego-centric grid containing obstacle information. To obtain this ego-centric grid, we discretize the map with a certain granularity. In this experiment, we used a 7×7 ego-centric grid that is able to define 49 area units surrounding



(a) (b) (c)

Fig. 4: Total reward per agent vs. the number of training episodes when the communication link is ideal for the number of agents = (a) 3; (b) 6; (c) 9.



(a) (b) (c)

Fig. 5: Explored spaces vs. the number of steps taken for 6 agents under communication failure probability p = (a) 0, (b) 0.3, (c) 0.6. Since the agents do not have any prior knowledge about the targets, explored spaces can reflect the efficiency of finding targets.

the agent. The *action* consists of the translation and rotation of the UAV. The *reward* is defined as a combination of multiple components including exploration rewards (+1), the target found reward (+5), and collision reward (-100). The rewards are designed in such a way that the agents could learn to explore more areas and avoid collisions. A well-trained model will learn to explore more areas while avoiding collisions with other agents as well as obstacles. At each time step, the agent will take an action and the environment will check if the new state reached after taking the action satisfies these conditions. For example, if the agent discovered a new area but collided with a teammate, then it will get both exploration and collision rewards, which sum up to -99. We disabled collision-avoidance algorithms to count the number of collisions to asses coordination performance.

Training and Testing Setup: Both low-level and high-level policy have a three-layer Multi-Layer Perceptron (MLP) structure with 256 units each. For the critic, the state and action will first go through a fully-connected layer with 64 units, and then will be concatenated and fed to a fully-connected layer with 256 units. The learning rate for actors and critics are 0.001 and 0.005, respectively. The two frameworks are both implemented in Pytorch [25], a deep-learning toolkit for Python. To add random exploration in the actors' behaviors, we add Ornstein-Uhlenbeck noise [26] to the output of the actor-network. A pre-trained Yolov4 [27] model is used as the detector for its good performance and high speed. As mentioned in Sect. I, in the experiments we do not adopt the centralized training and decentralized testing mechanism as others did. As a result, we will train the models with unreliable communication conditions and save them, and then use the trained model only for inference to generate the testing results. In the experiments, we used continuous space and action space; however, HMAAC also works for discrete action spaces with the help of the Gumbel-Softmax trick [28], which transforms the continuous output from the actor-network to a differentiable categorical distribution.

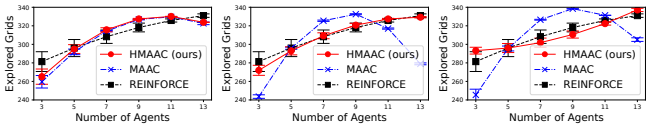
A. Bernoulli Communication Model

We evaluate here the performance of the proposed HMAAC under unstable communications and partial observability settings. We model packet loss as a Bernoulli process [5], [29], where we define a packet drop probability p. That is, with $p = 0$, the communication would be ideal while with $p = 1$ there will be no communication at all among the agents. We assume that, whenever UAV (A) wants to send a message to UAV (B), the packet containing the message has probability p of being dropped. For MAAC, this information could contain the states and actions of the UAV (A), which is needed when updating UAV (B). The experiments are conducted in the AirSim simulator, where we use a collection of 3D people models as victims and utilize the detection functionality provided by AirSim for victim detection. The obstacles' and the victims' locations are different for training and testing to ensure that the algorithm generalizes.

Rewards vs. Number of Episodes: To evaluate how these models learn during training, we plot the total reward averaged across the agents versus the number of episodes for different numbers of agents in Fig. 4, with a communication failure probability of $p = 0.4$ in this comparison. As the number of UAVs increases, the values of the rewards decrease because the number of victims and the search space is the same in these three scenarios, and a higher number of agents will lead to a decrease in the average reward. Furthermore, we can observe that HMAAC and MAAC converge to a good reward level in just a few episodes, while REINFORCE does not converge at all. This is because both HMAAC and MAAC can do a good job in a multi-agent scenario; conversely, for decentralized algorithms like REINFORCE, there is neither communication nor coordination, therefore performance degrades rapidly.

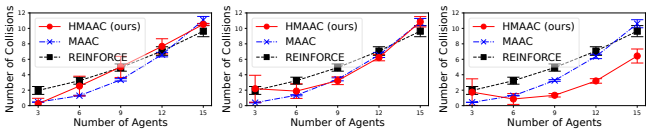
Exploration Speed: To see how the algorithms perform in terms of team exploration in different communication scenarios, we discretize the search space into a 20×20 grid and plot the cumulative number of explored grids in Fig. 5. Since the agents do not have any prior knowledge about the environment or the targets, the number of explored grids can reflect the efficiency of the target search system. It can be observed that, as the communication failure probability increases, the performance of HMAAC improves. This is because the agents can still maintain a decent global knowledge of the network from previous steps during disconnection.

Exploration Performance vs. Number of Agents: To show the scalability of the proposed framework, we discretize the search space into a 20×20 grid and vary the number of UAVs deployed in the SAR mission. The results are shown in Fig. 6. It can be observed that, as the number of agents increases, MAAC will experience a diminishing return. This is due to the fact that MAAC is flat-structured and thus can only handle a low number of agents (8 and 9 for p equal to 0.3 and 0.6, respectively), implying that once



(a) (b) (c)

Fig. 6: Explored spaces vs. the number of agents for communication failure probability p = (a) 0; (b) 0.3; (c) 0.6. Since the agents do not have any prior knowledge about the targets, explored spaces can reflect the efficiency of finding targets.



(a) (b) (c)

Fig. 7: Number of inter-agent collisions occurred per episode vs. number of agents for p=(a) 0; (b) 0.3; (c) 0.6.

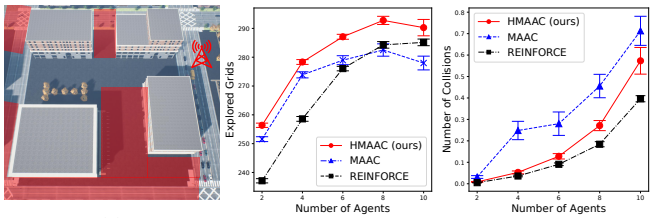
the number of agents "saturates", the performance of MAAC starts dropping.

Coordination Performance: One important metric to measure how well the coordination is achieved in multi-agent missions is the number of inter-agent collisions. Fig. 7 shows the results for different communication conditions. We observe that, as the number of agents increases, the number of collisions also increases, which is expected since more agents will lead to a greater chance of collisions due to increased coordination challenges. Moreover, we also notice that, as the communication condition deteriorates, the inter-collisions in HMAAC framework do not increase like in MAAC and REINFORCE; the former, in fact, shows less robustness to communication loss, while the latter cannot use coordination to avoid collisions.

B. Shadow Zones

Besides modeling packet dropping as a Bernoulli process, we also evaluate the proposed HMAAC against other models in a shadow zone scenario, i.e., where there are multiple buildings in the search area; we assume that these buildings can block communications and create shadow zones, in which agents will not be able to communicate with others. A demonstration of this scenario is shown in Fig. 8a. When UAVs enter these zones, they will lose communication with the CP and can only perform local updates without any being able to acquire global information. This scenario is evaluated in AirSim with buildings and targets scattered around.

Exploration Performance vs. Number of Agents: To show the performance in terms of explored areas, we plot the number of explored grids during testing against the number of agents in Fig. 8b. It can be observed that HMAAC achieves the best performance among these three models as the number of agents increases. Moreover, at one point, the performance of MAAC will drop and becomes worse than decentralized REINFORCE due to the saturation of the



(a) (b) (c)

Fig. 8: (a) Illustration of the shadow-zone scenario. There are four buildings in this scenario, and the Coordination Point (CP) is located on the right side of the buildings. Due to building blockage, areas with red colors cannot be reached by the CP, and therefore these zones are shadow zones; (b) Explored grids across all agents against the number of agents; (c) Number of collisions vs. number of agents.

number of agents. When the number of agents reaches 8, REINFORCE outperforms MAAC because the bad coordination due to disconnection in MAAC causes negative effects on the group performance.

Collision Performance: We plot the number of inter-agent collisions against the number of agents in Fig. 8c, which shows that the REINFORCE achieves the best performance in terms of number of collisions. However, in Fig. 8b, it is not the best model in terms of exploration, which means that, although REINFORCE has learned not to collide, it is not good at exploring the environment due to the lack of coordination.

V. Conclusion and Future Work

We presented a distributed aerial search and rescue framework with a hierarchical multi-agent actor-critic. To relax the interdependency among agents, we introduced a high-level policy, and the low-level agents are assumed to be conditionally independent given the coordination action. We deployed the high-level policy on network Coordination Points (CPs) and the actors and critics on Unmanned Aerial Vehicles (UAVs). To evaluate the performance of the proposed framework, we tested HMAAC in two scenarios, (a) unstable communication and (b) shadow zone. HMAAC was compared with MAAC and decentralized REINFORCE. The results showed that HMAAC is more scalable and robust to communication loss since it outperforms the other two models when the number of agents is large and when the communication condition is bad.

In the future, we plan to (i) extend the proposed framework to decentralized versions, where the communications are restricted to n-hop links, and remove the need for communicating to the CP every time; (ii) explore the on-demand communication mechanism to reduce the communication overhead further. (iii) adopt a federated update protocol, where each agent will copy the high-level policy to their local storage and perform joint optimization for the actor, critic, and the copy. Then, after a predefined interval, the local agents will upload their copies to the CP for merging in order to aggregate the knowledge;

REFERENCES

[1] V. Sadhu, T. Misu, and D. Pompili, "Deep multi-task learning for anomalous driving detection using can bus scalar sensor data," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2038–2043, IEEE, 2019.

[2] V. Sadhu, S. Zonouz, and D. Pompili, "On-board deep-learning-based unmanned aerial vehicle fault cause detection and identification," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5255–5261, IEEE, 2020.

[3] V. Sadhu, C. Sun, A. Karimian, R. Tron, and D. Pompili, "AerialDeepSearch: Distributed Multi-Agent Deep Reinforcement Learning for Search Missions," in *IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, (Delhi NCR), pp. 1–9, dec 2020.

[4] W. Li, H. Chen, B. Jin, W. Tan, H. Zha, and X. Wang, "Multi-agent path finding with prioritized communication learning," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 10695–10701, 2022.

[5] C. Igoe, R. Ghods, and J. Schneider, "Multi-agent active search: A reinforcement learning approach," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 754–761, 2021.

[6] C. Zhang and V. Lesser, "Coordinating multi-agent reinforcement learning with limited communication," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pp. 1101–1108, 2013.

[7] J. Foerster, I. A. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Advances in neural information processing systems*, pp. 2137–2145, 2016.

[8] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.

[9] C. Sun, T. Jiang, S. Zonouz, and D. Pompili, "Fed2kd: Heterogeneous federated learning for pandemic risk assessment via two-way knowledge distillation," in *2022 17th Wireless On-Demand Network Systems and Services Conference (WONS)*, pp. 1–8, IEEE, 2022.

[10] N. Basilico and F. Amigoni, "Exploration strategies based on multi-criteria decision making for searching environments in rescue operations," *Autonomous Robots*, vol. 31, no. 4, pp. 401–417, 2011.

[11] Y. Mei, Y.-H. Lu, C. G. Lee, and Y. C. Hu, "Energy-efficient mobile robot exploration," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 505–511, IEEE, 2006.

[12] D. C. Schedl, I. Kurmi, and O. Bimber, "An autonomous drone for search and rescue in forests using airborne optical sectioning," *Science Robotics*, vol. 6, no. 55, p. eabg1188, 2021.

[13] E. Lygouras, N. Santavas, A. Taitzoglou, K. Tarchanidis, A. Mitropoulos, and A. Gasteratos, "Unsupervised human detection with an embedded vision system on a fully autonomous uav for search and rescue operations," *Sensors*, vol. 19, no. 16, p. 3542, 2019.

[14] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, "Lsar: Multi-uav collaboration for search and rescue missions," *IEEE Access*, vol. 7, pp. 55817–55832, 2019.

[15] G. A. Cardona and J. M. Calderon, "Robot swarm navigation and victim detection using rendezvous consensus in search and rescue operations," *Applied Sciences*, vol. 9, no. 8, p. 1702, 2019.

[16] J. P. Queralta, J. Taipalmaa, B. C. Pullinen, V. K. Sarker, T. N. Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, and T. Westerlund, "Collaborative multi-robot systems for search and rescue: Coordination and perception," *arXiv preprint arXiv:2008.12610*, 2020.

[17] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), pp. 6382–6393, Curran Associates Inc., 2017.

[18] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *FSR*, 2017.

[19] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.

[20] J. Delmerico, E. Mueggler, J. Nitsch, and D. Scaramuzza, "Active autonomous aerial exploration for ground robot path planning," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 664–671, 2017.

[21] C. Sampedro, A. Rodriguez-Ramos, H. Bavle, A. Carrio, P. de la Puente, and P. Campoy, "A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques," *Journal of Intelligent & Robotic Systems*, vol. 95, no. 2, pp. 601–627, 2019.

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

[23] D. Han, W. Boehmer, M. Wooldridge, and A. Rogers, "Multi-agent hierarchical reinforcement learning with dynamic termination," in *Pacific Rim International Conference on Artificial Intelligence*, pp. 80–92, Springer, 2019.

[24] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*, pp. 621–635, Springer, 2018.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.

[26] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2015.

[27] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[28] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[29] Y. Zhang, V. Paxson, S. Shenker, and L. Breslau, "The stationarity of internet path properties: Routing, loss, and throughput," tech. rep., Citeseer, 2000.