Heterogeneous Policy Networks for Composite Robot Team Communication and Coordination

Esmaeil Seraj, *Student Member, IEEE*, Rohan Paleja, *Student Member, IEEE*, Luis Pimentel, Kin Man Lee, Zheyuan Wang, Daniel Martin, Matthew Sklar, John Zhang, *Graduate Student Member, IEEE*, Zahi Kakish, and Matthew Gombolay, *Member, IEEE*

***Abstract*—High-performing human–human teams learn intelligent and efficient communication and coordination strategies to maximize their joint utility. These teams implicitly understand the different roles of heterogeneous team members and adapt their communication protocols accordingly. Multiagent reinforcement learning (MARL) has attempted to develop computational methods for synthesizing such joint coordination–communication strategies, but emulating heterogeneous communication patterns across agents with different state, action, and observation spaces has remained a challenge. Without properly modeling agent heterogeneity, as in prior MARL work that leverages homogeneous graph networks, communication becomes less helpful and can even deteriorate the team's performance. In the past, we proposed heterogeneous policy networks (HetNet) to learn efficient and diverse communication models for coordinating cooperative heterogeneous teams. In this extended work, we extend HetNet to support scaling heterogeneous robot teams. Building on heterogeneous graph-attention networks, we show that HetNet not only facilitates learning heterogeneous collaborative policies, but also enables end-to-end training for learning highly efficient binarized messaging. Our empirical evaluation shows that HetNet sets a new state-of-the-art in learning coordination and communication strategies for heterogeneous multiagent teams by achieving an 5.84% to 707.65% performance improvement over the next-best baseline across multiple domains while simultaneously achieving a 200× reduction in the required communication bandwidth.**

***Index Terms*—Heterogeneous robot teaming, multirobot systems, neural graph-based communication.**

Manuscript received 20 December 2023; revised 22 April 2024; accepted 18 June 2024. Date of publication 22 July 2024; date of current version 12 August 2024. This work was supported in part by the Naval Research Lab under Grant N00173-21-1-G009, in part by the Lockheed Martin Corporation under Grant GR0-0000509, in part by the Office of Naval Research (ONR) under Grant N00014-19-1-2076, and in part by the Laboratory Directed Research and Development program at Sandia National Laboratories. This article was recommended for publication by Associate Editor N. Bezzo and Editor P. Robuffo Giordano upon evaluation of the reviewers' comments. *(Corresponding author: Rohan Paleja.)*

Esmaeil Seraj, Rohan Paleja, Luis Pimentel, Kin Man Lee, Zheyuan Wang, Daniel Martin, Matthew Sklar, and Matthew Gombolay are with the Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: rpaleja3@gatech.edu).

John Zhang is with the Carnegie Mellon University, Pittsburgh, PA 15213 USA.

Zahi Kakish is with the Sandia National Laboratories, Albuquerque, NM 87123 USA.

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2024.3431829, provided by the authors.

Digital Object Identifier 10.1109/TRO.2024.3431829

I. Introduction

HIGH-PERFORMING human–human teams benefit from communication to build and maintain shared mental models to improve team effectiveness [1], [2], [3]. Information sharing is key in building team cognition and enables teammates to cooperate to successfully achieve shared goals [2], [4], [5]. Further, typical communication patterns across human teams widely differ based on the task or role the human assumes [6]. The field of multiagent reinforcement learning (MARL) [7] has sought to develop agents that autonomously learn coordination and communication strategies to emulate high-performing human–human teams [8], [9], [10], [11], [12]. Yet, these approaches have fallen short in properly modeling heterogeneity and communication overhead in teaming [13], [14], [15], [16].

Heterogeneity in robots' design characteristics and their roles are introduced to leverage the relative merits of different robots and their capabilities [17], [18], [19], [20], [21], [22]. We define a heterogeneous robot team as a group of cooperative agents that are capable of performing different tasks and may have access to different sensory information. We categorize robot agents with similar state, action, and observation spaces in the same *class*. While multirobot teams can benefit from such heterogeneity through complementarity and combining different capabilities to accomplish complex tasks, communicating is not straightforward as agents do not "speak" the same language in such a heterogeneous setting as different classes of agents may need to process received information differently given their context. We consider scenarios in which agents have different action-spaces and observation inputs from the environment (i.e., due to different sensors) or may not even have access to any observation input (i.e., lack of sensors, broken or low-quality sensors), creating a dependency between sensor-constrained agents on agents with strong sensing capabilities. As such, in heterogeneous, composite robot teams, communication can be considered a necessity rather than a luxury to improve collaboration performance. In our prior work, Seraj et al. [23] developed an end-to-end heterogeneous graph-attention (HetGAT) architecture, heterogeneous policy networks (HetNet), for MARL that facilitates learning efficient, heterogeneous communication protocols among cooperating agents to accomplish a shared task. This framework was able to jointly accomplish modeling heterogeneity within heterogeneous robot teams and learning sender–receiver specific communication patterns. Through this architecture, we achieved a 200$\times$ reduction in the number of communicated bits per round of communication over baselines while also setting a new state-of-the-art (SOTA) in team performance for coordinating composite teams.

In this extension, we are concerned with learning cooperative behaviors for heterogeneous robot teams *at scale*, enabling our approach to learn in large task configurations with sparse reward schemes (i.e., global as opposed to individualized), transfer learned knowledge to novel team compositions and task configurations, and achieve policy robustness to noise-degraded communication channels. As we focus on coordinating heterogeneous robot teams across a variety of team compositions, it can become challenging to design complementary and fine-tuned agent-specific reward schemes that lead to high-performing cooperative behavior. Further, the number of composite team permutations possible is numerous, making *rigid MARL* frameworks that are parametric in the number of agents less useful. For example, if we are able to learn a coordination policy for a heterogeneous team consisting of two autonomous ground vehicles (AGVs) and three unmanned aerial vehicles (UAVs) within a search-and-rescue problem, MARL frameworks that require learning a brand new policy (and communication strategy) for *three* AGVs and *one* UAV can be cost-inefficient. Instead, we require *adaptive* and *scalable* MARL frameworks that can transfer learned knowledge to support heterogeneous robot teams across a variety of configurations. Finally, heterogeneity in robot teams naturally leads to differences in communication hardware, such as transmitter or receiver design, which can induce various forms of noise in the communication protocol. This noise can cause erroneous data to be received, ultimately resulting in a degraded policy for decision-making. Scalable communication protocols must be robust to various forms of noise to maximize the utility of each available agent.

In this article, we go beyond our prior work [23] and create a novel MARL framework to support generating coordination policies for heterogeneous robot teams at scale. Beyond this prior work, we first design a multiagent proximal policy optimization (PPO) variant to improve policy learning feasibility in difficult task configurations, improving both sample efficiency and training speed. Second, we create a scalable architecture that is invariant to the number of robot agents in each class, the size of the environment, and the size of an agent's observation space, allowing for a MARL framework that can transfer knowledge from smaller task configurations to larger, more difficult configurations. Third, we provide a framework for learning coordination policies under HetNet considering noise-degraded communication channels, addressing an important real-world consideration of multiagent robot teams. In this extended version, we present the following contributions.

1) We extend HetNet to develop a multiagent heterogeneous PPO (MAH-PPO) framework to learn ~~class-wise~~ cooperation policies where agents of a composite team learn inter- and intraclass cooperation policies to efficiently communicate and coordinate their actions with heterogeneous teammates. We find that optimizing with PPO enables us to learn policies in larger-scale domains where learning was inefficient and difficult due to increased heterogeneity within the multirobot team.
2) We modify our graph-based architecture to include a fully convolutional network (FCN), resulting in invariability to 1) the size of the environment and 2) the number of agents in the team. This facilitates learning a policy that can be *transferred* to large-scale domains, achieving a 389.51% improvement in sample efficiency, and a 80.63% decrease in run-time required to achieve high-performance.
3) We conduct an empirical evaluation across several domains and team compositions. We present evidence that are evolved variant of HetNet sets a new SOTA in learning emergent heterogeneous cooperative policies by achieving at least 5.84%–707.65% performance improvement over baselines in small domain configurations and 32.5%–1161.35% performance improvement across larger-scale domain configurations.
4) We extend our framework to support noisy communication channels and show that HetNet is robust to data perturbations. With a poor signal-to-noise ratio (3.24 db) under additive white Gaussian noise, HetNet performs just 4.57% worse than with a perfect communication channel. We introduce two additional formulations for communication loss (range-based and type-based) to allow a more thorough assessment of robustness to noise within a heterogeneous robot team setting.
5) We demonstrate our algorithm on physical robots on a physical, multirobot testbed.

The rest of this article is organized as follows. In Sections II and III, we introduce related work and preliminaries, respectively. In Section IV, we introduce our graph-based architecture that supports modeling heterogeneous teams of various size (contribution 2). In Section V, we describe our training and execution procedure, including our MAH-PPO learning algorithm (contribution 1). In Section VI, we describe our evaluation environments, baselines, empirical findings, and ablation study across degraded communication channels (contribution 3 and 5). In Section VII, we present a real-world robot demonstration (contribution 5) of our algorithm within a heterogeneous robot team. Finally, Section VIII concludes this article.

II. Related Work

MARL With Communication—Recently, the use of communication in MARL has been shown to enhance the collective performance of learning agents in cooperative MARL problems [8], [13], [14], [24], [25], [26], [27], [28], [29], [30], [31], [32]. DIAL [33], and CommNet [25] displayed the capability to learn discrete and continuous communication vectors, respectively. While DIAL considers the limited-bandwidth problem, neither of these approaches are readily applicable to composite teams or capable of performing attentional communication. TarMAC [14] achieves targeted communication through an attention mechanism, which improves performance compared to prior work. Nevertheless, TarMAC requires high-bandwidth message-passing channels, and its architecture is reported to perform poorly in capturing the topology of interaction [26].

SchedNet [34] explicitly addresses the bandwidth-related concerns. However, in SchedNet, agents learn how to schedule themselves for accessing the communication channel rather than learning the communication protocols from scratch. Furthermore, across several MARL works discussed in this section, including [14], [24], [25], [35], a key limitation is that these frameworks required an individualized reward scheme to scale to larger task configurations. Individualized reward schemes, while assisting with the credit assignment problem in MARL [36], can learn suboptimal coordination strategies in heterogeneous robot teams. A more effective, albeit more difficult-to-learn strategy is to use a shared team reward scheme, enabling agents to learn cohesive policies that achieve the composite team's goal, rather than learning policies that benefit the skill of a particular class of agent. Furthermore, in our MARL approach, we explicitly address the heterogeneous communication problem where agents learn diverse communication protocols and intermediate language representations to use among themselves for cooperation. Our model enables agents to perform attentional communication and send limited-length digitized messages through class-specific encoder-decoder channels, addressing the limited-bandwidth issues.

MARL With Graph Neural Networks (GNN)—Prior work on MARL has sought to utilize GNNs to model a communication structure among agents [37]. Deep Graph Network (DGN) [38] represents dynamic multiagent interaction as a graph convolution to learn cooperative behaviors. This seminal work in MARL demonstrates that a graph-based representation substantially improves performance. In [39], an effective communication topology is proposed by using hierarchical GNNs to propagate messages among groups and agents. G2ANet [26] proposed a game abstraction method combining a hard and a soft-attention mechanism to dynamically learn interactions between agents. More recently, MAGIC [35] introduced a scalable, attentional communication model for learning a centralized scheduler to determine when to communicate and how to process messages through graph-attention networks. While these prior work have successfully modeled multiagent interactions, they are not designed to address heterogeneous teams directly. HetNet, on the other hand, is designed to capture the heterogeneity among agents and learn an efficient shared language across agents with different action and observation spaces to improve cooperativity.

Heterogeneity in Multiagent Systems—In [40], several types of heterogeneity induced by agents of different capabilities are discussed. As opposed to homogeneous teams, the diversity among agents in heterogeneous teams makes it challenging to hand-design intelligent communication protocols. In [41], a control scheme is hand-designed for a heterogeneous multiagent system by modeling the interaction as a leader–follower system. More recently, HMAGQ-Net [42] utilized GNNs and Deep Deterministic Q-network (DDQN) to facilitate coordination among heterogeneous agents (i.e., those with different state and action spaces). Going beyond this prior work, we build our HetNet model based upon an actor-critic framework and generalize the problem formulation for state-, action-, and observation-space heterogeneities. Moreover, HetNet facilitates learning efficient binary representations of states as an intermediate language among agents of different types to improve cooperativity.

Noisy Communication in Multiagent Systems—Prior work investigating the effects of noise in MARL has primarily focused on noise within the environment [43], [44], where observations may not accurately represent the true state of the environment. While these works demonstrate methods to handle errors from noisy agent observations, they assume a perfect communication channel between the agents. Wu et al. [45] and Karabag et al. [46] proposed planning frameworks for multiagent systems with intermittent or full loss of communication but do not consider noisy data in the communication channel. MARL frameworks such as [47], [48], [49] explicitly model communication noise through a binary symmetric channel or additive Gaussian noise, but none of them address both heterogeneity and bandwidth limitations simultaneously. HetNet, in contrast, can learn near-optimal strategies under noisy and bandwidth-limited communication for heterogeneous teams, traits that are required for robot teams as they begin to scale.

III. Preliminaries

Founding on a standard partially observable MDP (POMDP) [50], we formulate a new problem setup termed as multiagent heterogeneous POMDP (MAH-POMDP), which can be represented by a 9-tuple $\langle \mathcal{C}, \mathcal{N}, \{\mathcal{S}^{(i)}\}_{i\in\mathcal{C}}, \{\mathcal{A}^{(i)}\}_{i\in\mathcal{C}}, \{\Omega^{(i)}\}_{i\in\mathcal{C}}, \{\mathcal{O}^{(}i)\}_{i\in\mathcal{C}}, r, \mathcal{T}, \gamma\rangle$. $\mathcal{C}$ is a set of all available agent classes in the composite robot team, and the index $i \in \mathcal{C}$ shows the agent class. $\mathcal{N} = \sum_{\langle i\in\mathcal{C}\rangle} N^{(i)}$ is the total number of collaborating agents, where $N^{(i)}$ represents the number of agents in class i. $\{\mathcal{S}^{(i)}\}_{i\in\mathcal{C}}$ is a discrete joint set of state-spaces. For each class-dependent state-space, $\mathcal{S}^{(i)}$, we have $\mathcal{S}^{(i)} = \left[s_t^{i_1}, s_t^{i_2}, \ldots, s_t^{i_{N^{(i)}}}\right]$, where $s_t^{i_j}$ represents the state-vector of robot agent j of the i-th class, at time t. $\{\mathcal{A}^{(i)}\}_{i\in\mathcal{C}}$, is a discrete joint set of action-spaces. For each state-dependent action-space, $\mathcal{A}^{(i)}$, we have $\mathcal{A}^{(i)} = \left[a_t^{i_1}, a_t^{i_2}, \ldots, a_t^{i_{N^{(i)}}}\right]$, forming the joint action-vector of agents of class i at time t. $\{\Omega^{(i)}\}_{i\in\mathcal{C}}$ is similarly defined as the joint set of observation-spaces, including class-specific observations. $\gamma \in [0, 1)$ is the temporal discount factor for each unit of time, and $\mathcal{T}$ is the state transition probability density function.

At each timestep, t, each agent [1], j, of the ith class can receive (if the observation input is enabled for class i) a partial observation $o_t^{i_j} \in \Omega^{(i)}$ according to some class-specific observation function $\{\mathcal{O}^{(i)}\}_{i\in\mathcal{C}} : o_t^{i_j} \sim \mathcal{O}^{(i)}(\cdot|\bar{s})$. If the environment observation is not available for agents of class i, agents in the respective class will not receive any input from the environment (e.g., lack of sensory inputs). Regardless of receiving an observation or not, at each time, t, each robot agent, j, of class i, takes an action, $a_t^{i_j}$, forming a joint action vector $\bar{a} = \left(a_t^{1_1}, a_t^{1_2}, \ldots, a_t^{i_1}, \ldots, a_t^{i_j}\right)$. When agents take the joint action $\bar{a}$, in the joint state $\bar{s}$ and depending on the next joint state, they receive an immediate reward, $r(\bar{s}, \bar{a}) \in \mathbb{R}$, shared by

[1] By "agent" we are mainly referring to a "robot agent."

all agents and regardless of their classes. Our objective in (1) is to learn near-optimal policies, per existing agent-class, to solve the MAH-POMDP by maximizing the total expected, discounted reward accumulated by agents over an infinite horizon, i.e.,

$$\pi(\bar{s}) = \underset{\pi(\bar{s})\in\Pi}{\arg\max}\, \mathbb{E}_{\pi(\bar{s})}\left[\sum_{k=0}^{\infty}\gamma^k r_{t+k}|\pi(\bar{s})\right]. \quad (1)$$

A. Policy Gradient Methods

Policy gradient methods are an approach to RL that utilize function approximation, in which each agent j has a policy, $\pi_\theta^j(a|s)$, parameterized by θ, that specifies which action, a, to take in each state, s, to maximize the expected future discounted reward. Policy gradient methods apply gradient ascent to the actor's parameters, θ, based on a gradient estimate of the expected return in (1). By the policy gradient theorem [51], the expected reward maximization, $J(\theta)$, is maximized via (2), where a_t^j and o_t^j are the action and observation of agent j, respectively.

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta^j}\left[\nabla_\theta \log \pi_\theta^j(a_t^j|o_t^j)\hat{A}_t(o_t^j, a_t^j)\right]. \quad (2)$$

The *advantage* function, $\hat{A}_t(o_t^j, a_t^j) = Q(o_t^j, a_t^j) - V^\phi(o_t^j)$, measures how much better the action is relative to the default action of the current policy [52], where $Q(o_t^j, a_t^j)$ is the state-action value function approximated by the total discounted rewards, and $V^\phi(o_t^j)$ is the state-value function, approximated according to a critic network parametrized by ϕ.

B. Proximal Policy Optimization

PPO was introduced to improve training stability, sample efficiency, and performance in single-agent RL tasks [52]. PPO achieves better training stability by constraining the size of gradient updates to the policy, such that the current policy does not change too much from the previous policy. This is achieved by the clipped surrogate objective in (3), where $r_t(\theta)$ is the probability ratio between the current and old policy before updating (4), and ϵ is a hyperparameter defining the clip range

$$J^{\text{CLIP}}(\theta) = \mathbb{E}\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \quad (3)$$

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}. \quad (4)$$

In addition, an entropy term, $J^{\text{S}}(\theta) = c_1 S(\pi_\theta(s_t))$, is added to ensure sufficient exploration, where S is the policy entropy and c_1 is the entropy coefficient hyperparameter. PPO also performs multiple steps of mini-batch gradient updates, leading to higher sample efficiency.

C. Graph Neural Networks

GNNs are a class of deep neural networks that capture the structural dependency among nodes of a graph via message-passing between the nodes, where each node aggregates feature vectors of its neighbors to compute a new feature vector [38], [53], [54]. The canonical feature update procedure via graph convolution operator can be shown as $\bar{h}'_j = \sigma\left(\sum_{k\in N(j)} \frac{1}{c_{jk}}\omega\bar{h}_k\right)$, where $\bar{h}'_j$ is the updated feature vector for node j, $\sigma(.)$ is the activation function and, ω represents the learnable weights. $k \in N(j)$ includes the immediate neighbors of node j, where k is the index of neighbor, and c_{jk} is the normalization term, which depends on the graph structure. A common choice of c_{jk} is $\sqrt{|N(j)N(k)|}$. In an L-layer aggregation, a node j's representation captures the structural information within the nodes that are reachable from j in L hops or fewer. However, the fact that c_{jk} is structure-dependent can impair generalizability of GNNs when scaling the graph's size. Thus, a direct improvement is to replace c_{jk} with attention coefficients, α_{jk}, computed via (5). In (5), $\bar{W}_{\text{att}}$ is the learnable weight, $\|$ represents concatenation, and $\sigma'(.)$ is the LeakyReLU nonlinearity. The Softmax function is used to normalize the coefficients across all neighbors k, enabling feature-dependent and structure-free normalization [55], [56].

$$\alpha_{jk} = \underset{k}{\text{softmax}}\left(\sigma'\left(\bar{W}_{\text{att}}^T\left[\omega\bar{h}_j \,\|\, \omega\bar{h}_k\right]\right)\right). \quad (5)$$

IV. Method

In this section, we first present an overview of the communication problems and constraints considered in our work. We then describe how to construct a heterogeneous graph given a problem state and present the building block layer of our model, which we refer to as the HetGAT layer. Next, we develop a novel preprocessing module that enables our architecture to be invariant to the number of robot agents in each class, the size of the environment, and the size of an agent's observation space. Afterward, we discuss our binarized encoder–decoder communication channel that accounts for the heterogeneity of messages passed among agents, while digitizing messages for effective utilization of bandwidth. Finally, we cover the logistics of utilizing HetGAT layers to assemble our HetNet of arbitrary depth.

A. Communication Problem Overview

In this work, we are concerned with the problem of coordinating a robot team via fostering direct communication among interacting agents. We consider MARL problems wherein multiple agents interact in a single environment to accomplish a task that is of a cooperative nature. We are particularly interested in scenarios in which the agents are heterogeneous in their capabilities, meaning agents can have different state, action, and observation spaces in forming a composite team. To collaborate effectively, agents must share messages that express their observations under a centralized training and distributed execution (CTDE) paradigm [34], [36].

In learning an end-to-end communication model, we take a series of problems and constraints into consideration: 1) heterogeneous messages, where agents of different classes have different action and observation spaces, resulting in different interpretations of sent/received messages; 2) attentional and scalable communication protocols, such that agents incorporate

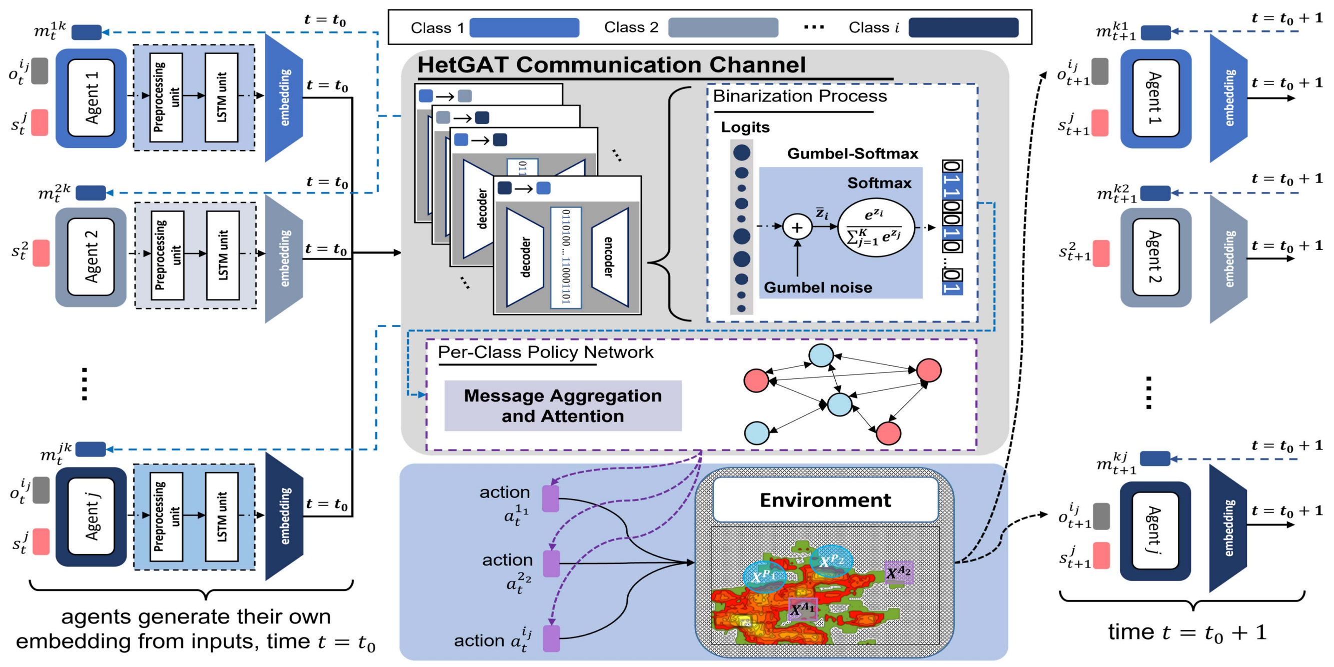


Fig. 1. Overview of our multiagent heterogeneous attentional communication architecture in a CTDE paradigm. At each time point $t = t_0$, each agent j of class i generates a local embedding from its own inputs, by passing its input data through class-specific preprocessing units and an LSTM cell. Each agent then sends the embedding to a class-specific encoder–decoder networks to generate a binarized message, m_t^{jk}, from its local neighbor k. The message information is then decoded and aggregated via learned attention coefficients by the receiving agent to compute the action probabilities as its policy output.

attention coefficients depending on the agent/class they are communicating with for coordinating with teammates in any arbitrary team sizes; 3) learning communication models for low-size, -weight, and -power (Low-SWAP) systems, where due to limited communication bandwidth (CB), robot agents must learn to communicate in a highly efficient shared intermediate "language" (e.g., limited-length binarized messages); 4) limited-range communications, where agents can only exchange messages when they are within close proximity.

B. Heterogeneous Communication Model

GNNs previously used in MARL operate on homogeneous graphs to learn a universal feature update and communication scheme for all agents [26], [35], [38], [39], which fail to explicitly model the heterogeneity among agents. We instead cast the cooperative MARL problem into a heterogeneous graph structure, and propose a novel HetGAT network capable of learning diverse communication strategies based on agent classes. Compared to homogeneous graphs, a heterogeneous graph can have nodes and edges of different types that can have different types of attributes. This advantage greatly increases a graph's expressivity and enables straightforward modeling of complicated, composite teams.

Given our MAH-POMDP formulation in the preliminaries section, we directly model each agent class $i \in \mathcal{C}$ as a unique node type. This approach allows agents to have different types of state-space content, $\mathcal{S}^{(i)}$, as input features according to their classes, as well as enabling different types of action spaces, $\mathcal{A}^{(i)}$. Communication channels between agents are modeled as directed edges connecting the corresponding agent nodes. When two agents move to a close proximity of each other such that those agents fall within communication range, we add bidirectional edges to allow message passing between them. We use different edge types to model different class combinations of the sender and receiver agents to allow for learning heterogeneous communication protocols and intermediate representations.

Accordingly, we present an overview of our multiagent heterogeneous attentional communication architecture in Fig. 1. At each time, t, the features of each agent (i.e., each node of the heterogeneous graph) are generated through a class-specific feature preprocessor. We utilize separate modules to preprocess an agent's state, $s_t^{i_j}$, and observation, $o_t^{i_j}$, since depending on the agent's class, the environment observation input may not be available. Each preprocessing module contains a FCN [57] unit followed by an LSTM unit to enable reasoning about both spatial and temporal information, further discussed in the preprocessing tensorized representations Section IV-C and visualized in Fig. 2. As shown in Fig. 1, the generated embeddings are then passed into a HetGAT communication channel, including a class-specific encoder–decoder network and a Gumbel-Softmax [58] unit to generate a binarized message, m_t, for an agent, j.

C. Preprocessing Tensorized Representations

In this evolved version of HetNet, we are concerned with scaling heterogeneous robot teams to multiagent domains with larger environment sizes and larger team sizes, varying

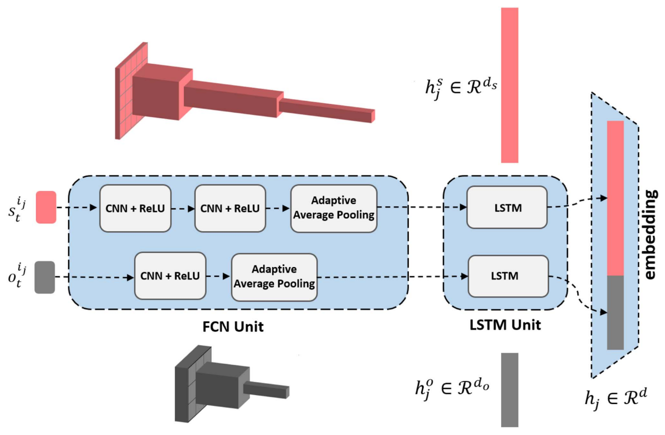


Fig. 2. Visualization of our proposed preprocessing architecture. Separate FCN Units, for state and observation inputs, generate fixed-length embeddings from size-varying tensorized input representations. The hidden embeddings of the following LSTM units are combined, generating a fixed-length embedding for the HetGAT communication channels.

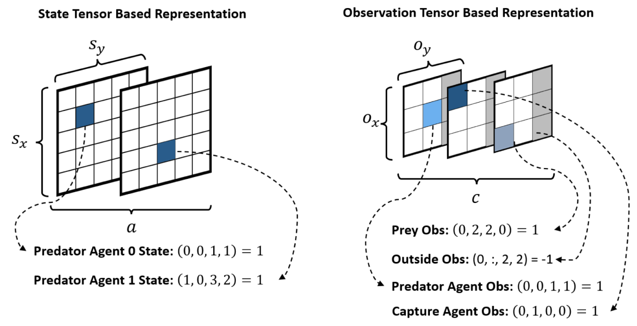

State Tensor Based Representation
s_y
s_x
a
Predator Agent 0 State: $(0,0,1,1) = 1$
Predator Agent 1 State: $(1,0,3,2) = 1$
Observation Tensor Based Representation
o_y
o_x
c
Prey Obs: $(0,2,2,0) = 1$
Outside Obs: $(0, :, 2, 2) = -1$
Predator Agent Obs: $(0,0,1,1) = 1$
Capture Agent Obs: $(0,1,0,0) = 1$

Fig. 3. Proposed tensorized input representations for an agent's state and observation, utilizing multidimensional tensors with dimensions (i, c, s_x, s_y) and (i, c, o_x, o_y). The visualization shows the encoding for a 5×5 PCP environment with a 3×3 observation range.

in its compositionality, to allow for both effective learning within a single configuration and transferring knowledge to larger configurations that require significantly more exploration. Prior works have designed communicative MARL frameworks with input feature representations, namely *vectorized* inputs, using fixed-length one-hot encodings to encode an agent's state and observation. These representations are specific to a domain configuration (e.g., environment size and observation range) and do not effectively capture spatial information. As such, the learned policies designed for vectorized inputs cannot be deployed to domain configurations with larger environments and larger team sizes, as this would require additional parametrization to handle increased input dimensions. This inability to be nonparametric to task configuration size disqualifies prior work from utilizing transfer learning methods to warm-start learning in large-scale domain configurations, where learning can be much more difficult and computationally expensive.

We improve the existing HetNet policy architecture to support utilizing a variable-sized, tensorized *feature representation of an agent's state and observations.* This enables us to transfer coordination policies learned in smaller domain configurations in order to warm-start learning in larger, more challenging domain configurations. We define the state and observation tensorized representations with the multidimensional tensors (i, c, s_x, s_y), and (i, c, o_x, o_y), respectively. Across both tensors, the i dimension indexes an agent's state or observation in the environment. The cardinality of this dimension is determined by the number of agents in a particular class, as agents may not have observations in our heterogeneous setting. Furthermore, the channel dimension, c, corresponds to an agent's class. For the state tensor, the cardinality of this dimension is set to one, as only one channel is necessary to encode the state. For the observation tensor, the cardinality of this dimension is determined by the number of classes in the heterogeneous domain. The last two dimensions encode spatial information about the state or observation of the robot agent, namely positions in the environment. For the state tensor, the cardinality of the s_x and s_y dimensions correspond to the size of the environment. For the observation tensor, the cardinality of the o_x and o_y dimensions correspond to an agent's observation range, and positional encodings are relative to the agent's reference frame. As a brief example, as depicted in Fig. 3, the state representation of the location of Agent 1 of Class 0, at grid location $(3, 2)$ in a 5×5 environment, would have the following tensor value: $(1, 0, 3, 2) = 1$.

We design preprocessing modules that can process tensorized input representations of varying dimensions in the size of the environment and the number of agents in the team. The architecture of a preprocessing module, shown in Fig. 2, consists of an FCN unit to reason about spatial information and an LSTM unit to reason about temporal information. Tensorized state and observation inputs are processed by separate preprocessing modules, as an agent may or may not have input observations, as detailed in our MAH-POMDP formulation in Section III. The output of the LSTM unit is denoted by $h_j^s \in \mathbb{R}^{d_s}$ and $h_j^o \in \mathbb{R}^{d_o}$, for each preprocessing module, where d_s and d_o are the feature dimensions of state and observation embeddings, respectively. These embeddings are concatenated such that the input feature embedding to the HetGAT communication channel is denoted as $h_j = [h_j^s \| h_j^o] \in \mathbb{R}^d$, where $d = d_s + d_o$. The purpose of our FCN architecture is twofold: 1) the convolutional neural network (CNN) layers are able to better encode spatial and class-type information by processing multiple, grid-like channels, and 2) the adaptive average pooling layer is able to generate a fixed-length embedding, regardless of varying size tensorized representation inputs. Together, these properties enable us to learn better policies in large-scale heterogeneous domains and enable transferring a learned policy to new domain configurations without the need to add additional parametrization as required by prior work.

Several works in MARL [59], [60], [61], [62] utilize tensor-based representations for representing multiple agents within a grid, as these representations can be utilized across tasks. The closest related work to ours is [61], which also provides results transferring a policy trained on a small domain with few agents to a larger domain with more agents. However, their MARL formulation uses a fully observable MDP, while

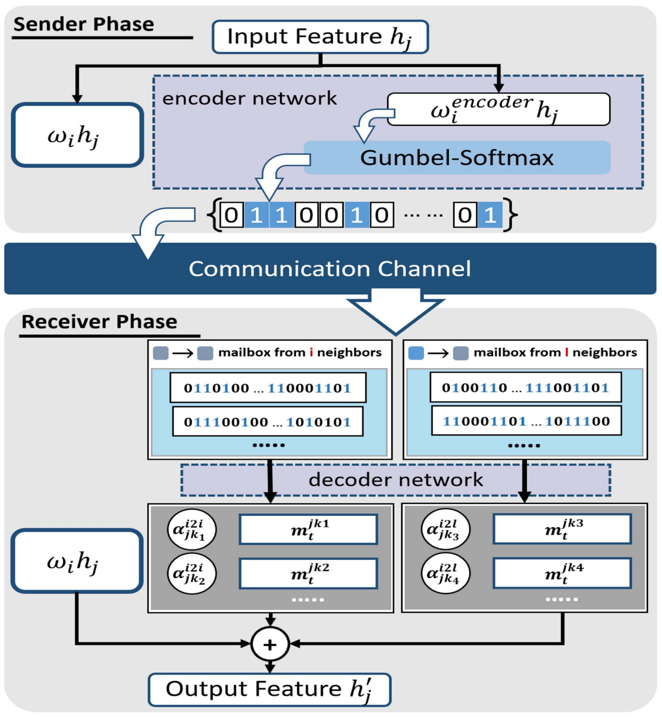


Fig. 4. Sender and receiver phases of the feature update process in a HetGAT layer for one agent, j, of class i.

ours is a POMDP augmented with message communication between agents to mitigate partial observability. We note while this formulation limits our approach to domains where state and observation information can be represented as tensors, this space represents a variety of domains, including applications ranging from adversarial team versus team battles within the StarCraft multiagent challenge [63] to drone-fleet navigation for radiological inspection [64], where each drone utilizes voxel-based representations.

D. Binarized Communication Channels

The feature update process in a HetGAT layer is conducted in two steps: *per-edge-type* message passing followed by *per-node-type* feature reduction. When modeling multiagent teams, we reformulate the computation process into two phases: a *sender* phase and a *receiver* phase. Fig. 4 shows the computation flow during the sender and receiver phases for an agent, j, of class i.

During the sender phase, the agent, j, of class $i \in \mathcal{C}$, processes its input feature, h_j, using a class-specific weight matrix, $\omega_i \in \mathbb{R}^{d' \times d}$, where d and d' are the input and output feature dimensions, respectively. The agent also transforms h_j into the assigned message dimension using a class-specific encoder, $\omega_i^{enc} \in \mathbb{R}^{n \times d}$, where n is the communication channel bandwidth. Next, we leverage a universal binarization process utilizing Gumbel-Softmax to convert the message into 0 s and 1 s for all classes as an efficient, intermediate language. The binarized message is then sent to neighboring agents. We note that, a class-specific encoder entails a peredge-type communication mechanism, meaning that the communication encoders learn to encode the embedding information according to the classes of both the sender and the receiver agents. The meaning of this class-specific encoded message is then determined at the class-specific decoder, again based on the receiver agent's class. Subsequently, the message decoded by each agent is directly integrated as part of the input space, along with the agent's state observation, to update the policy.

During the receiver phase, robot agent, j, of class i, processes all the received messages using a class-specific decoder, $\omega_i^{dec} \in \mathbb{R}^{d' \times n}$. Next, for each type of communication edge that an agent is connected to, the HetGAT layer computes per-edge-type aggregation result by weighing received messages, along the same edge-type with normalized attention coefficients, α^{edgeType}. The aggregation results are then merged with the agent's own transformed embedding, $\omega_i h_j$, to compute the output features. The feature update formula for an agent is shown in (6), where j and k are agent indexes and, $i, l \in \mathcal{C}$ are class indexes; such that, $i2l$ is an $edgeType$ and means "from class i to class l". m_t^{jk} is the decoded message computed by (7) and, $\Delta_l(j)$ include agent j's neighbors that belong to class l

$$\text{Class}(i) : \bar{h}_j' = \sigma\left(\omega_i \bar{h}_j + \sum_{l \in \mathcal{C}} \sum_{k \in N_l(j)} \alpha_{jk}^{i2l} m_t^{jk}\right) \tag{6}$$

$$m_t^{jk} = \omega_i^{dec}(\text{GumbelSoftmax}(\omega_l^{enc} h_k)). \tag{7}$$

Note that we have $l = i$ for intraclass communication. When computing attention coefficients in a heterogeneous graph, we adapt (5) into (8) to account for heterogeneous channels

$$\alpha_{jk}^{i2l} = \underset{k}{\text{softmax}}\left(\sigma'\left(\bar{W}_{att}^T\left[\omega_i \bar{h}_j \,\|\, \omega_{i2l} \bar{h}_k\right]\right)\right). \tag{8}$$

To stabilize the learning process, we adapt the multihead extension of the attention mechanism [55] to fit our heterogeneous setting. We use L independent HetGAT (sub-)layers to compute node features in parallel and then merge the results by concatenation operation for each multihead sublayer in HetNet except for the last layer, which employs averaging. As a result, each type of communication channel is split into L independent, parallel subchannels.

E. Heterogeneous Policy Network (HetNet)

At each timestep, t, a HetGAT layer corresponds to one round of message exchange between neighboring agents and feature update within each agent. By stacking several HetGAT layers, we construct the HetNet model that utilizes multiround communication to extract high-level embeddings of each agent for decision-making. For the last HetGAT layer in HetNet, we set each agent's output feature dimension to the size of its action-space, specific to its class, i. Then, for each agent node, we add a Softmax layer on top of its output to obtain a probability distribution over actions that can be used for action sampling, resulting in class-wise stochastic policies. Accordingly, the computation process of each agent's policy remains local for distributed execution.

V. Training and Execution

Our prior work with HetNet [23] utilized a multiagent heterogeneous actor–critic (MAHAC) formulation to learn policies across several grid-world domains, varying in complexity, with a maximum dimensionality of 5×5. However, deploying HetNets at scale requires 1) an effective learning algorithm that is sample efficient and able to learn in sparselyrewarded environments, and 2) the ability to model realistic perturbations to communication signals.

A. Multiagent Heterogeneous PPO (MAH-PPO)

Motivated by the pitfalls of MAHAC in scaling to larger domains, we sought to create a learning algorithm for heterogeneous settings that is more sample-efficient (reducing computation time) and performs well in multiagent settings. As such, given the recent success of PPO in multiagent settings, we extend MA-PPO [65] and present our MAH-PPO framework that directly accounts for heterogeneity in multiagent scenarios by learning class-wise coordination policies. This advancement allows us to learn high-performance policies under increased heterogeneity and size. We assign one policy per existing class, $\pi^i \in \{\Pi\}^{\mathcal{C}}$, each of which is parameterized by θ^i. We utilize a *centralized critic* value function V^ϕ, parameterized by ϕ, which takes as input the global state, including the agents' joint-state $\bar{s}_t$, and environment states. The trained actor network on the heterogeneous graph contains one set of learnable weights per agent class, which can be distributed to individual agents in the execution phase. During training, a centralized critic with access to global state input is shared by all agents, regardless of class, and can be disabled during decentralized execution following standard practice in CTDE paradigms.

To create MAH-PPO, we modify PPO's clipped surrogate objective [see (3)] to account for heterogeneity in the multiagent setting. In doing so, we obtain the per-class multiagent clipped surrogate objective in (9), where $|B|$ is the batch size of trajectories, $r^j_{\theta^i}$ is the per-class probability ratio of an agent j with a policy of class i, $\hat{A}^j_t$ is the advantage estimate of agent j, and ϵ is a hyperparameter defining the clip range

$$J^{\text{CLIP}}(\theta^i) = \frac{1}{|B|\mathcal{N}^{(i)}} \sum_{\tau \in B} \sum_{j=1}^{\mathcal{N}^{(i)}} \sum_{t=1}^{T} \min\left(r^j_{\theta^i}\hat{A}^j_t, \text{clip}(r^j_{\theta^i}, 1-\epsilon, 1+\epsilon)\right). \tag{9}$$

The per-class probability ratio in (10) measures the probability ratio of class i's policy, where $\bar{a}^{i_j}_t$ and $\bar{o}^{i_j}_t$ represent the joint actions taken and joint observations received (if applicable for class i) by agents at time, t, respectively. In addition, $\bar{m}_t$ represents the message-vector received by agent j from its neighbors

$$r^j_{\theta^i} = \frac{\pi_{\theta^i}\left(\bar{a}^{i_j}_t | \bar{o}^{i_j}_t, \bar{m}_t\right)}{\pi_{\theta^i_{\text{OLD}}}\left(\bar{a}^{i_j}_t | \bar{o}^{i_j}_t, \bar{m}_t\right)}. \tag{10}$$

The advantage estimates are computed via the GAE [52] method, where δ^j_t is agent j's temporal difference (TD) residual with discount γ, and $V^\phi(\bar{s}_t)$ is the centralized critic, parameterized by ϕ, with global state input $\bar{s}_t$

$$\hat{A}^j_t = \sum_{l=0}^{T} (\gamma\lambda)^l \delta^j_{t+l} \tag{11}$$

$$\delta^j_t = r_t + \gamma V^\phi(\bar{s}_{t+1}) - V^\phi(\bar{s}_t). \tag{12}$$

In addition, we add a per-class entropy term, in (14) to ensure exploration, where S is the policy entropy and c_1 is the entropy coefficient

$$J^{\text{S}}(\theta^i) = c_1 \frac{1}{|B|\mathcal{N}^{(i)}} \sum_{\tau \in B} \sum_{j=1}^{\mathcal{N}^{(i)}} \sum_{t=1}^{T} S\left(\pi_{\theta^i}(\cdot|\bar{o}^{i_j}_t, \bar{m}_t)\right). \tag{13}$$

Putting together (9)–(13), the policy for each class, π^i, is updated via gradient descent to maximize the MAH-PPO objective as follows:

$$J(\theta^i) = J^{\text{CLIP}}(\theta^i) + J^{\text{S}}(\theta^i). \tag{14}$$

Given the improvements in the learning ability of our MAH-PPO formulation, we are able to learn under a shared reward task setting where credit assignment is much more difficult compared to our prior work, especially as we increase the complexity of the domain. Motivated to explore these challenges and following prior work [65], we train HetNet with shared rewards. However, we note that HetNet can technically be trained with individual and class-specific rewards under our MAH-POMDP formulation in Section III. We provide further implementation details regarding our MAH-PPO implementation and hyperparameters in Section II-B of the supplementary.

B. Scalability Via Transfer

Learning high-performance MARL coordination policies in large-scale domain configurations is difficult due to the increasing joint state-action space that agents must explore, resulting in an increasingly complex credit-assignment problem [36] and a large variance of policy gradients during training [66]. Moreover, training MARL algorithms in large-scale domains has practical implications, including prolonged training times and the need for increased high-performance computational resources. In this section, we describe our approach for addressing these challenges by combining HetNet's ability to process varying-size tensorized input representations with transfer learning.

In our approach, we first train a HetNet policy on a simple domain configuration with a small environment size and a small number of agents. Training in this configuration ensures that we quickly achieve a high-performing communication-coordination policy with the ability to accomplish the task. We train a policy in this domain configuration until convergence before directly transferring to a domain configuration with a larger environment size and increased number of agents. We can transfer our policy in an end-to-end fashion, as our preprocessing modules accept size-varying tensorized inputs, without

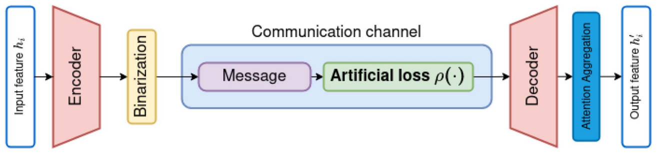


Fig. 5. Visualization of modification to the communication channel for artificial loss.

requiring additional parametrization. By directly transferring knowledge learned in small-scale domains, we are able to effectively and efficiently address the complexities of learning diverse large-scale communication protocols and intermediate language representations. This facilitates faster learning performance and enables policies to learn in large domain configurations where learning policies from scratch can fail.

C. Lossy Communication

A ubiquitous assumption of multiagent communication frameworks is the reliability of interagent communication. In practice, however, communication channels can be subject to signal perturbations from hardware failures, electromagnetic noise, and adversarial manipulation (e.g., signal jamming). While noise in digital communication has been explored extensively in works such as [67], [68], [69], the study of noisy communication in MARL is scarce. In this section, we describe how we modify the communication channel of HetNet during training to account for noisy communication and introduce formulations for three different types of noisy channels. This artificial-manipulation communication scheme can be easily applied to other MARL communication frameworks to evaluate the performance of their learned policies when subject to imperfect communication channels.

To simulate such loss in our framework, we introduce artificial bit perturbation in the communication channel between agents. Fig. 5 shows a high-level overview of where artificial loss is introduced in our communication pipeline. The input features of a sender agent k, h_k are processed by its class encoder and binarized to produce a compressed and efficient message for transmission. Then, a form of loss is applied just prior to the decoding of the message by the receiving agent. By doing so, we can accurately represent where the majority of real-world noise is introduced, i.e., during the transmission of the message in the communication channel. The decoded message computed in (7) is adapted into (15) to account for a noisy channel

$$m_t^{jk} = \omega_i^{\text{dec}}(\rho(\text{GumbelSoftmax}(\omega_l^{\text{enc}} h_k))). \tag{15}$$

In this equation, ρ represents an arbitrary model of the communication channel that outputs a noisy message. We note that ρ is nondifferentiable due to the stochasticity of communication noise, thus prohibiting backpropagation during training. In our experiments, we evaluate three types of noisy communication channels over HetNet-Binary with tensor-based representation of agent observations. The first is a communication channel with static white Gaussian noise, where the noise level between all agents is fixed throughout a given experiment. This is the most basic scenario that allows us to measure the impact of noise as its intensity increases. The second noise model incorporates the distance between communicating agents, enabling an evaluation of our method's sensitivity to dynamically changing noise. The third model is a type-based noise model with different noise levels conditioned on the classes of the two communicating agents. Heterogeneous agents, by definition, have diverse traits, which include the electrical hardware for communication signals. Type-based noise allows us to capture noise that can result from heterogeneity and measure its impact on our architecture. In all three implementations, the noisy channel ρ is parameterized by the bit error rate (BER) p_b as follows:

$$\rho_{X\sim\text{Bern}(p_b)}(b_i) = \begin{cases} b_i & \text{if X = 0} \\ 1 - b_i & \text{if X = 1} \end{cases}, \forall b_i \in m_e. \tag{16}$$

In this equation, m_e is the encoded message from the GumbelSoftmax that performs message binarization, as indicated in (15).

Additive White Gaussian Noise (AWGN): In an AWGN channel with binary phase-shift keying modulation, the BER p_b is given in [67] by

$$p_b = Q\left(\sqrt{\frac{2\varepsilon_b}{N_0}}\right) \tag{17}$$

$$Q(x) = P[\mathcal{N}(0,1) > x] = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-\frac{t^2}{2}}dt. \tag{18}$$

$\frac{\varepsilon_b}{N_0}$ is the signal-to-noise (SNR) ratio per bit. The SNR is negatively correlated with the BER, i.e., decreasing the SNR increases the chances of a bit flip and vice versa.

Range-Based Noise With AWGN: By the inverse square law, the intensity of a signal is inversely proportional to the square of the distance from the signal source. This property is expressed in the range-based noise model by scaling the signal strength with respect to the distance between two communicating agents. Equation (17) is adapted into (19) to reflect this property

$$p_b = Q\left(\sqrt{\frac{2\varepsilon_b}{N_0}\cdot\frac{1}{D^{jk}}}\right). \tag{19}$$

In (19), D^{jk} is the Euclidean distance between two agents j and k in our 2D-grid space environment. Each pair of agents is exposed to dynamic noise that changes with each time step.

Type-Based Noise With AWGN: To account for the diverse types of transmitter, receiver, and modulation schemes in digital communication design, we introduce a general noise model to capture signal attenuation that can occur between different types of agents based on their communication scheme. Equation (17) is adapted into (20) to model these deficiencies that can arise from the heterogeneity of the agents

$$p_b = Q\left(\sqrt{\gamma^e\cdot\frac{2\varepsilon_b}{N_0}}\right), 0 \le \gamma^e \le 1. \tag{20}$$

In (20) γ^e is a tunable signal attenuation constant that is specific for each edge type in the communication framework. Note that all $\gamma^e = 1$ is equivalent to the base noise model with AWGN.

VI. Empirical Evaluation

In this section, we discuss our evaluation environments, baselines that we compare our proposed formulation against, and results. We provide a full implementation of our approach https://github.com/CORE-Robotics-Lab/HetNet/tree/hetnet-ppo.

A. Evaluation Environments

We evaluate the utility of HetNet against several baselines in two cooperative and heterogeneous MARL domains that require learning collaborative behaviors. These domains consist of two-class perception-action team of agents, where different classes of agents may have different observation and action spaces. In this section, we describe the key objectives and the differences between the two domains. For complete descriptions and further details of the dynamics of each environment, please refer to Section I of the supplementary document.

Predator-Capture-Prey (PCP) [23] – For the first heterogeneous domain, we modify the homogeneous predator–prey [24] environment to create a new heterogeneous environment, which we refer to as predator–capture–prey (PCP) [23], to include a composite team. In PCP, there are two classes of agents: *predator* and *capture* agents. The *predator* class of agents have the ability to perceive the environment and have the goal of finding the prey with limited vision. We term these agents "Perception" agents. The *capture* class of agents have the goal of locating the prey *and* capturing it with an additional *capture–prey* action in their action-space, and importantly, do not have any observation inputs (e.g., lack of scanning sensors). We term these agents "Action" agents. We define three configurations for this domain, varying the environment and team sizes, as in [24]. The baseline configuration (*small*) for the PCP environment is a size 5×5 grid-world with 2 Perception (P) and 1 Action (A) agents. In this configuration, the perception agent's vision is limited as they only observe a 1×1 grid around them. Likewise, we define the *medium* configuration as a size 10×10 grid-world with 3P and 2 A agents, and the *large* configuration as a 20×20 grid-world with 6P and 4 A agents. In these configurations, perception agents can observe a 3×3 grid around them.

FireCommander (FC) [70]—In the second heterogeneous domain, the FireCommander [70], two classes of *perception* and *action* agents must collaborate as a composite team to extinguish a propagating firespot. At each timestep, the firespot propagates to a new location according to the FARSITE [71] model, while the previous location is still on fire. All firespots are initially hidden to agents and need to be discovered before being extinguished. As such, *perception* agents are tasked to scan the environment to detect the firespots while *action* agents (no observation inputs) are required to move and extinguish a firespot that has been discovered by a *perception* agent before. Note that since firespots propagate, both *perception* and *action* agents need to continue to explore the map and collaborate until all firespots are extinguished. We define the domain configurations for the FC environment the same as in PCP, however, perception agents have increased observation range as they can observe a 3×3 grid around them in both *small* and *medium* configurations.

B. Baselines

We benchmark our framework against four end-to-end communicative MARL baselines: 1) CommNet [25], 2) IC3Net [24], 3) TarMAC [14] and, 4) MAGIC [35]. We note that, for all four baselines we directly pulled the respective authors' publicly available code-bases and hyperparameters for training. In addition, to further evaluate the effectiveness of MAH-PPO, we compare against HetNet trained with MAHAC as in prior work [72]. All baseline methods are trained using a default vectorized input representation. For fair comparison, we modify each baseline's critic to accept the environment-provided global state, and train the baselines using both shared and individual rewards.

C. Results and Discussion

In this section, we empirically validate the performance of our frameworks across several heterogeneous teaming domains against the introduced baselines. Across each domain, we seek to minimize the number of steps taken to accomplish the search-and-rescue problem.

1) Performance Comparison to Baselines: We compare HetNet's performance against other baseline methods on the *small* domain configurations of the PCP [23] and FC [70] domains. Fig. 6 depicts the average steps taken ($\pm$ standard error) by each method across episodes as training proceeds in each configuration of both domains. In both domains, PCP and FC, HetNet outperforms all baselines, trained with either shared (S) or individual (I) reward, by converging to a more efficient coordination policy (i.e., fewer steps taken).

Furthermore, we evaluate the learned coordination policies at convergence by each of the baselines in both domains. For fair comparison, we randomly generate a set of 100 initial conditions for the positions of agents and prey or fire locations, for each domain configuration, and evaluate each method with these same initial conditions. The results of this evaluation are presented in Fig. 7, where the reported results are the average [$\pm$ standard error (SE)] steps taken from 100 evaluation trials. As shown, in both the PCP and FC domains, HetNet outperforms all baselines trained with either a shared or individual reward scheme. We highlight that only HetNet demonstrates the ability to learn a high-performing policy in the *small* configuration of the FC domain, which is of higher complexity than the PCP domain. HetNet's superior performance can be explained by its ability to handle heterogeneity, while baseline methods struggle as they are not designed for this purpose.

We also note that baseline methods perform better when trained with shared reward, compared to individual reward within the same configuration. This can be attributed to shared reward enabling agents to learn policies that achieve the composite $team's$ goal rather than learning policies that benefit the skill of a particular class of agent. Our results indicate that this can be particularly important for heterogeneous teaming, as training with individual rewards can lead to learning suboptimal policies.

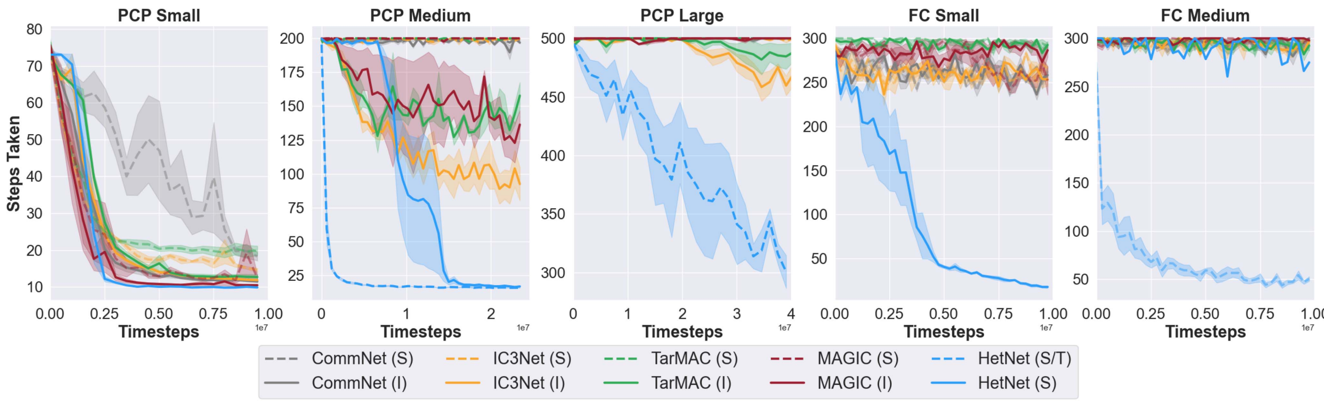


Fig. 6. Learning curves of training in the PCP and FC domains. Shown is average steps taken (± SE) by each method across episodes and three different random seeds as training proceeds. Baseline methods are trained with shared (S) and individual rewards (I). HetNet PPO (S) is trained with shared reward, and HetNet PPO (S/T) is trained with shared reward via transfer. HetNet outperforms all baselines in both domains in performance at convergence and sample efficiency.

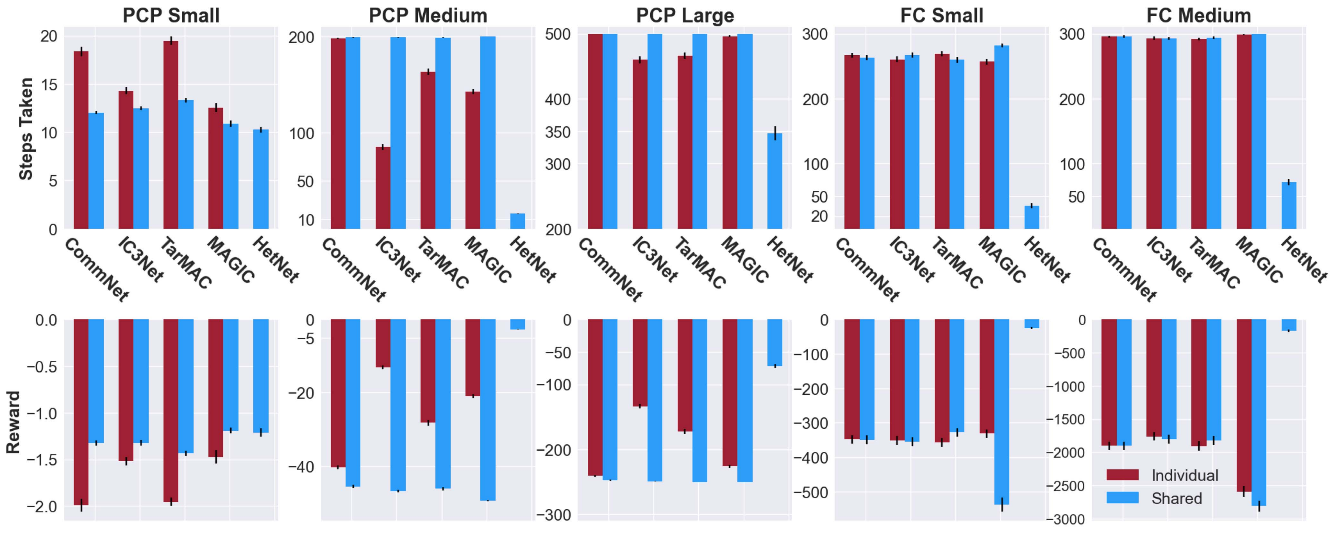


Fig. 7. Evaluation results across domain configurations. Reported results are averaged (± standard error (SE)) from 100 evaluation trials with different random-seed initializations. For all tests, the final training policy at convergence is used for each method. As shown, HetNet outperforms all baselines in both heterogeneous domains.

Across both domains, HetNet achieves a 5.1% to 707.65% performance improvement over other baselines, trained with either share or individual rewards, in the *small* domain configurations. The heterogeneous policies learned by our model set the SOTA for learning challenging cooperative behaviors for composite teams.

2) Performance Comparison to HetNet Under MAHAC: In addition to other end-to-end communicative MARL baselines, we compare our method against our prior work that trains HetNet under MAHAC [72] in both the small and medium configurations of PCP and FC. For a fair comparison to HetNet with MAH-PPO, we train HetNet with MAHAC under a shared reward scheme. We evaluate the trained policies under the same evaluation conditions described in Section VI-C1 and highlight our results in Fig. 8. We observe that MAH-PPO outperforms MAHAC in both domains and across both configuration scales. In the small domain configuration, we observe that MAH-PPO

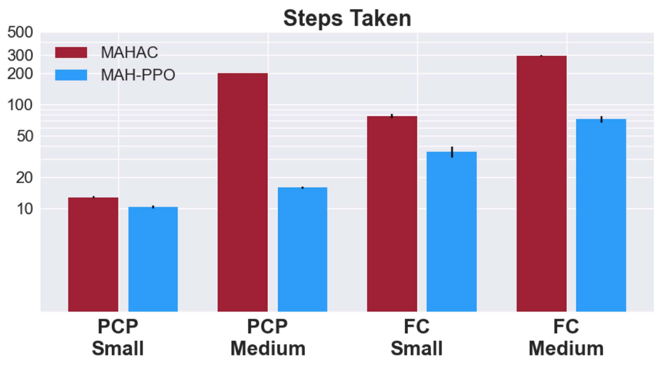
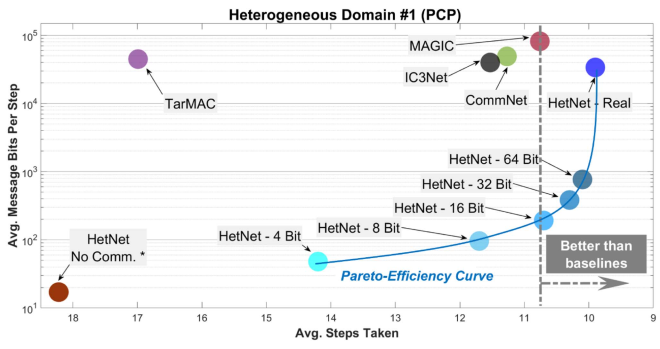
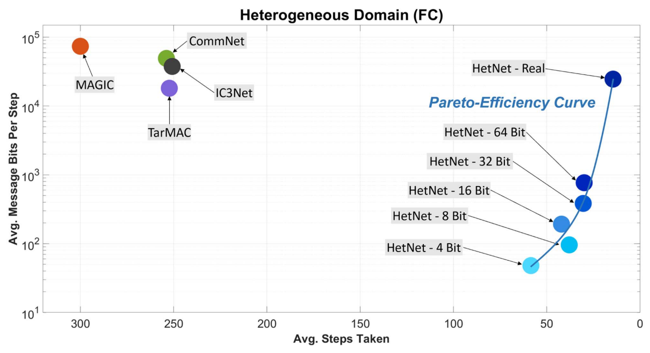


Fig. 8. Performance comparison between HetNet trained with MAHAC in prior work [72] and HetNet trained with MAH-PPO. As shown, HetNet trained with MAH-PPO outperforms HetNet trained with MAHAC across domain configurations.



(a)



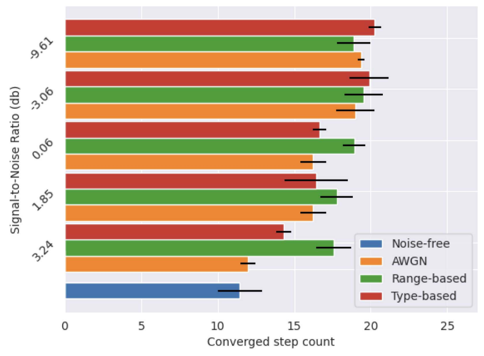
(b)

Fig. 9. Results of experiments assesing CB as communicated bits per round of communication. HetNet facilitates binarized messages among agents which requires significantly less CB as compared to real-valued baselines. (a) CB versus performance results in in PCP domain as shown in [23]. (b) CB versus performance results in the FC domain.

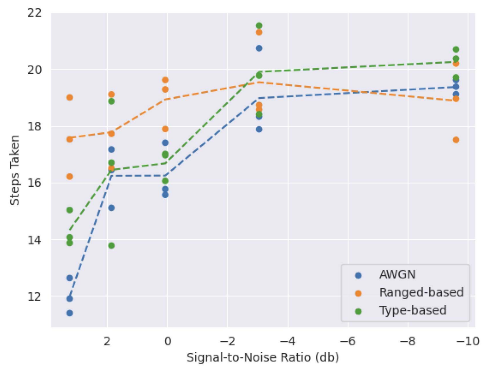
achieves a 23.83% to 122.10% performance improvement over MAHAC. Most notably, we observe that MAH-PPO is most beneficial in the more complex medium domain configurations, as it achieves a 307.7% to 1166.09% performance improvement over MAHAC. Our results highlight the importance of designing advanced optimization methods, such as MAH-PPO, to enable learning in complex heterogenous domains, where prior formulations fail to learn high-performing coordination policies.

To supplement our empirical comparison to MAHAC, in Section II-B of the supplementary, we highlight several implementation improvements in training HetNet under MAH-PPO that enable feasible training in large-scale domains, such as PCP Large. In Section II-C of the supplementary, we compare the runtime performance of training HetNet with MAH-PPO and training with the implementation of MAHAC. We find that due to elongated training times, training HetNet with MAHAC in PCP large is intractable. With the implementation improvements, HetNet trained with MAH-PPO leads to a 98.68% decrease in experiment runtime.

3) Scalability: We conduct experiments to evaluate the scalability of our framework, against other baselines, to larger environments and team sizes in the *medium* and *large* configurations of the PCP and FC domain. In Fig. 6, we observe that HetNet, trained from scratch, converges much quicker than other baseline methods trained on the *medium* configuration of the PCP domain. As shown in Fig. 7, HetNet PPO achieves a 32.5% to 1161.35% performance improvement over other baselines, trained in the *medium* and *large* domain configurations, outperforming all other baselines trained with either shared (S) or individual (I) reward. Furthermore, we observe that in the *medium* and *large* configurations, baseline methods achieve higher performance when trained using individual rewards instead of shared rewards, as opposed to their performance on the *small* domain configurations. This might be attributed to individual reward enabling agents to learn policies that explore the larger scale domain better in a way that benefits their individual skills (e.g., perception agents first learn to find the prey and then learn how to communicate this information to action



(a)



(b)

Fig. 10. Results of experiments assessing the robustness of our architecture to communication noise. All results are averaged over three random seeds and experiments are performed in the *small* domain configuration of PCP. (a) Average steps taken across episodes at convergence under all noise conditions. All methods are trained with HetNet-PPO (S) and 16-bit messages. (b) Average steps taken across episodes as SNR decreases in *small* domain configuration of PCP.

agents). Moreover, training with shared reward makes the credit-assignment problem more difficult [36]. Despite this challenge, HetNet, trained with shared reward, is able to overcome the difficulty in exploration and converge to a much higher-performing policy than other baselines, displaying the proficiency of the

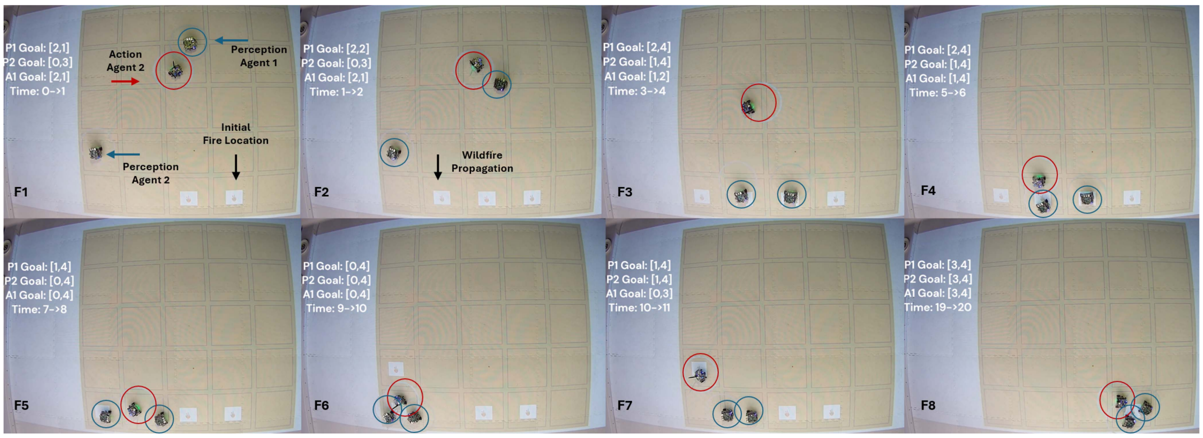


Fig. 11. Demonstration of our algorithm in the FireCommander domain on physical robots on the Robotarium platform. We highlight the different agents as well as the propagating wildfire through eight frames pulled from the demonstration video. In each frame, perception agents are circled in blue, while action agents are circled in red. We denote the goal location of each agent and the timestep of each frame on the top-right. The complete video and implementation details can be found in the supplementary material.

adapted architecture of HetNet and associated MAH-PPO formulation.

4) Transferability: We show the utility of our proposed architecture, with the augmentations described in Section IV-C, by scaling to both *medium* and *large* configurations of the PCP domain via transfer as described in scalability via transfer section. We pretrain a HetNet policy in the *small* domain configuration with increased vision (i.e., an agent's observation range is 3×3) to ensure that we pretrain observation preprocessing modules for the same observation range as specified in the *medium* domain configuration. We train this policy until convergence, transfer this learned policy to the more challenging *medium* domain configuration with more agents and increased dimensionality, and continue training. In Fig. 6, we observe that the HetNet policy trained after transfer in the PCP *medium* configuration is more sample efficient and converges much quicker than a policy trained from scratch. Notably, it takes approximately 15 million training samples for the HetNet policy trained from scratch to achieve less than 20 steps per episode, while the transferred HetNet policy can achieve the same performance in approximately 3 million training samples. This represents a 389.51% improvement in sample efficiency and an 80.63% decrease in runtime required to achieve a high-performing policy. We further leverage the transferability of our architecture and deploy a policy, trained until convergence in the *medium* configuration, to the PCP *large* configuration. In Fig. 6, we observe that our transferred policy is able to achieve higher performance than other baselines trained from scratch in the PCP *large* configuration. In addition, in Fig. 6, we observe the same benefits of transferability, as a HetNet policy trained to convergence in the FC *small* configuration is transferred to the FC *medium* configuration and is able to achieve higher performance than baselines trained from scratch. Our results demonstrate that our transferable architecture enables improved sample efficiency and enables learning in challenging domains where conventional training from scratch can fail.

5) Robustness: To assess the robustness of our architecture to communication noise, we perform a sensitivity analysis on HetNet when exposed to each of the three previously described noise models. We compare the performance of HetNet under the three noise models against HetNet with a noise-free channel to show the impact of noise on performance, as perfect communication serves as the lower bound on the number of steps that can be achieved. For all channel types, the network is trained with a tensorized input representation, shared reward, 16-bit binarized messages, and the same hyperparameters in the *small* configuration of PCP. We chose a reduced message dimension to demonstrate the effectiveness of our architecture while under bandwidth constraints and noise simultaneously. For type-based noise, we fixed the signal attenuation constant γ^e for edges between *predator* and *capture* agents to 0.8 to model communication deficiencies between different classes of agents. The average steps taken at convergence for all three noise models ($\pm$ standard error) over a range of SNR is shown in Fig. 10(a) and (b).

Under the AWGN noise model, we observe a small performance decrease when the SNR is high. The performance continues to decay exponentially as SNR decreases and plateaus as the noise power overwhelms the signal power (i.e., when the SNR is negative). The final performance with negative SNR approaches the performance of HetNet with no communication ($\approx$20 steps) as shown in [23]. With the range-based noise model, we observe a significant decrease in performance relative to the noise-free communication, even when the SNR is high. This indicates that our architecture is sensitive to high variability in noise during learning as the noise level changes based on the distance between agents. However, we note that the range-based noise condition with a positive SNR still outperforms scenarios with no communication. For type-based noise, we see that performance is similar to the AWGN condition with a slight decrease. This shows that our architecture is able to learn under noise introduced from the heterogeneity of composite teams.

In Fig. 10(b), we additionally show the performance of our architecture with noise across all seeds and a range of SNR. We observe more variance in the final converged performance as opposed to HetNet (S) in Fig. 6, which can be attributed to the binarization of messages, as similar variance can be observed with noise-free communication. In general, a higher SNR results in a decrease in the lower bound for average steps taken.

6) Communication Bandwidth: We compute the CB for HetNet (i.e., the number of bits communicated per round) over a range of message dimensionalities and show performance in the *small* domain configuration of FC in Fig. 9(b). The performance over CB follows a similar pareto-efficiency curve as shown in Fig. 9(a) from our prior work [23]. None of the baselines were able to converge in the FC domain. All HetNet-Binary configurations were able to achieve better performance than the baselines, demonstrating the effectiveness of our framework under bandwidth restrictions.

VII. Real-World Robot Demonstration

We present a demonstration of our algorithm emulating the *Small* FireCommander example with two perception agents and one action agent on the Robotarium platform, a remotely accessible swarm robotics research platform [73]. In Fig. 11, we display several frames (F1–F8) of a trajectory being executed in this scenario from a top–down perspective (not ego-centric to any agent), where perception agents are encircled in blue and the action agent is encircled in red. This demonstration showcases the feasibility of trajectories produced via HetNet on real robots and the learned policy's coordination strategy throughout the complex FC scenario. Notably, we observe that in frame 2 (F2), perception agent 2 (PA2) becomes aware of a fire location as the fire propagates in the environment. We observe that this information is successfully communicated to action agent 1 (AA1), which cannot reason about fire locations due to its lack of sensory observations, and perception agent 1 (PA1). By F4, we observe all agents have moved to the fire locations and can observe AA1 putting out fires in later frames, F5 and F6. In F7, we highlight the coordination policy's strategy as AA1 navigates to the previously discovered propagating fire location, and both perception agents explore new locations for fire spots. At F8, all agents converge at the last fire location before AA1 puts out the last fire. Once this action is taken, the episode terminates. We attach the full video of this demonstration in the supplementary material with annotations. Furthermore, we include additional implementation details of this demonstration in Section III of the supplementary document. We leave training MARL algorithms on real-world heterogeneous robots, which can require a large number of samples, to future work.

VIII. Conclusion

In this extended work, we create a scalable MARL architecture and associated learning algorithm to support heterogeneous robot team coordination, advancing the SOTA of MARL algorithms augmented with communication. We provide several enhancements to our prior work [23], including 1) the development of a novel MAH-PPO algorithm to support sample-efficient learning of coordination policies for heterogeneous robots, 2) an architecture enhancement to support transferability to different environment sizes, resulting in nonparametricity in the number of agents, and 3) a noise-degradation channel with three different paradigms of loss. We find that HetNet can outperform baselines across several domains, scale to domain configurations in which the baselines fail to learn, and develop robustness to noise-degraded communication channels.

Acknowledgment

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC (NTESS), a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration (DOE/NNSA) under contract DE-NA0003525. This written work is authored by an employee of NTESS. The employee, not NTESS, owns the right, title and interest in and to the written work and is responsible for its contents. Any subjective views or opinions that might be expressed in the written work do not necessarily represent the views of the U.S. Government. The publisher acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this written work or allow others to do so, for U.S. Government purposes. The DOE will provide public access to results of federally sponsored research in accordance with the DOE Public Access Plan.

References

[1] J. MacMillan, E. E. Entin, and D. Serfaty, "Communication overhead: The hidden cost of team cognition," in *Team cognition: Understanding the Factors That Drive Process and Performance*. Washington, DC, USA: American Psychological Association, 2004, pp. 61–82.

[2] J. E. Mathieu, T. S. Heffner, G. F. Goodwin, E. Salas, and J. A. Cannon-Bowers, "The influence of shared mental models on team process and performance," *J. Appl. Psychol.*, vol. 85, no. 2, 2000, Art. no. 273.

[3] E. Seraj, J. Y. Xiong, M. L. Schrum, and M. Gombolay, "Mixed-initiative multiagent apprenticeship learning for human training of robot teams," in *Proc. 37th Conf. Neural Inf. Process. Syst.*, 2023, pp. 35426–35440.

[4] E. Salas, T. L. Dickinson, S. A. Converse, and S. I. Tannenbaum, "Toward an understanding of team performance and training," in *Teams: Their Training and Performance*. New York, NY, USA: Ablex Publishing, 1992.

[5] S. G. Konan, E. Seraj, and M. Gombolay, "Iterated reasoning with mutual information in cooperative and Byzantine decentralized teaming," in *Proc. Int. Conf. Learn. Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=giBFoa-uS12

[6] H. Taylor, "The effects of interpersonal communication style on task performance and well being," Ph.D. dissertation, Univ. Buckingham, Buckingham, U.K, 2007.

[7] L. Busoniu, R. Babuka, and B. D. Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Systems, Man, Cybern. C. Appl. Rev.*, vol. 38, no. 2, pp. 156–172, Mar. 2008.

[8] C. Zhang and V. R. Lesser, "Coordinating multi-agent reinforcement learning with limited communication," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2013, pp. 1101–1108.

[9] C. Yu, X. Wang, and Z. Feng, "Coordinated multiagent reinforcement learning for teams of mobile sensing robots," in *Proc. 18th Int. Conf. Auton. Agents MultiAgent Syst.*, 2019, pp. 2297–2299.

[10] J. M. Catacora Ocana, F. Riccio, R. Capobianco, and D. Nardi, "Cooperative multi-agent deep reinforcement learning in soccer domains," in *Proc. 18th Int. Conf. Auton. Agents MultiAgent Syst.*, 2019, pp. 1865–1867.

[11] O. Vinyals et al., "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[12] C. Berner et al., "Dota 2 with large scale deep reinforcement learning," 2019, *arXiv:1912.06680*.

[13] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," in *Handbook of Reinforcement Learning and Control*. Berlin, Germany: Springer, 2021, pp. 321–384.

[14] A. Das et al., "Tarmac: Targeted multi-agent communication," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1538–1546.

[15] Y. Du et al., "Learning correlated communication topology in multi-agent reinforcement learning," in *Proc. 20th Int. Conf. Auton. Agents MultiAgent Syst.*, 2021, pp. 456–464.

[16] H. Mao, Z. Gong, Y. Ni, and Z. Xiao, "Accnet: Actor-coordinator-critic net for "learning-to-communicate" with deep multi-agent reinforcement learning," 2017, *arXiv:1706.03235*.

[17] H. Ravichandar, K. Shaw, and S. Chernova, "Strata: Unified framework for task assignments in large teams of heterogeneous agents," *JAAMAS*, vol. 34, no. 2, 2020, Art. no. 38.

[18] E. Seraj, Z. Wang, R. Paleja, M. Sklar, A. Patel, and M. Gombolay, "Heterogeneous graph attention networks for learning diverse communication," 2021, *arXiv:2108.09568*.

[19] E. Seraj, L. Chen, and M. C. Gombolay, "A hierarchical coordination framework for joint perception-action tasks in composite robot teams," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 139–158, Feb. 2022.

[20] E. Seraj, A. Silva, and M. Gombolay, "Safe coordination of human-robot firefighting teams," 2019, *arXiv:1903.06847*.

[21] R. Aragues, D. V. Dimarogonas, P. Guallar, and C. Sagues, "Intermittent connectivity maintenance with heterogeneous robots," *IEEE Trans. Robot.*, vol. 37, no. 1, pp. 225–245, Feb. 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:226526712

[22] E. Seraj, A. Silva, and M. Gombolay, "Multi-UAV planning for cooperative wildfire coverage and tracking with quality-of-service guarantees," *Auton. Agents Multi-Agent Syst.*, vol. 36, no. 2, 2022, Art. no. 39.

[23] E. Seraj et al., "Learning efficient diverse communication for cooperative heterogeneous teaming," in *Proc. Int. Conf. Adaptive Agents Multi-Agent Syst.*, 2022, pp. 1173–1182.

[24] A. Singh, T. Jain, and S. Sukhbaatar, "Individualized controlled continuous communication model for multiagent cooperative and competitive tasks," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=rye7knCqK7

[25] S. Sukhbaatar et al., "Learning multiagent communication with backpropagation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2244–2252.

[26] Y. Liu, W. Wang, Y. Hu, J. Hao, X. Chen, and Y. Gao, "Multi-agent game abstraction via graph attention neural network," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 7211–7218.

[27] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7254–7264.

[28] T. Chu, S. Chinchali, and S. Katti, "Multi-agent reinforcement learning for networked system control," in *Proc. 8th Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, Apr. 26–30, 2020. [Online]. Available: https://openreview.net/forum?id=Syx7A3NFvH

[29] X. Xu, R. Li, Z. Zhao, and H. Zhang, "Stigmergic independent reinforcement learning for multiagent collaboration," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 9, pp. 4285–4299, Sep. 2022.

[30] E. Seraj, "Embodied, intelligent communication for multi-agent cooperation," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 16135–16136.

[31] E. Pesce and G. Montana, "Improving coordination in small-scale multi-agent deep reinforcement learning through memory-driven communication," *Mach. Learn.*, vol. 109, pp. 1727–1747, 2020.

[32] E. Seraj and M. Gombolay, "Coordinated control of UAVs for human-centered active sensing of wildfires," in *Proc. Amer. Control Conf.*, 2020, pp. 1845–1852.

[33] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2137–2145.

[34] D. Kim et al., "Learning to schedule communication in multi-agent reinforcement learning," in *Proc. 7th Int. Conf. Learn. Representations*, New Orleans, LA, USA, May 6–9, 2019. [Online]. Available: https://openreview.net/forum?id=SJxu5iR9KQ

[35] Y. Niu, R. Paleja, and M. Gombolay, "Multi-agent graph-attention communication and teaming," in *Proc. 20th Int. Conf. Auton. Agents MultiAgent Syst.*, 2021, pp. 964–973.

[36] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proc. 32nd AAAI Conf. Artif. Intell., 30th Innov. Appl. Artif. Intell., 8th AAAI Symp. Educ. Adv. Artif. Intell.*, S. A. McIlraith and K. Q. Weinberger, Eds., New Orleans, LA, USA, Feb. 2–7, 2018, pp. 2974–2982. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17193

[37] D. Adjodah et al., "Communication topologies between learning agents in deep reinforcement learning," 2019. [Online]. Available: https://ifaamas.org/Proceedings/aamas2020/pdfs/p1738.pdf

[38] J. Jiang, C. Dun, T. Huang, and Z. Lu, "Graph convolutional reinforcement learning," in *Proc. 8th Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, Apr. 26–30, 2020. [Online]. Available: https://openreview.net/forum?id=HkxdQkSYDB

[39] J. Sheng et al., "Learning structured communication for multi-agent reinforcement learning," 2020, *arXiv:2002.04235*.

[40] M. Bravo, J. A. Reyes-Ortiz, J. Rodríguez, and B. Silva-López, "Multi-agent communication heterogeneity," in *Proc. Int. Conf. Comput. Sci. Comput. Intell.*, 2015, pp. 583–588.

[41] S. Xiong, Q. Wu, and Y. Wang, "Distributed coordination of heterogeneous multi-agent systems with output feedback control," in *Proc. IEEE Int. Conf. Unmanned Syst. Artif. Intell.*, 2019, pp. 106–111.

[42] D. D. R. Meneghetti and R. A. D. C. Bianchi, "Towards heterogeneous multi-agent reinforcement learning with graph neural networks," 2020, *arXiv:2009.13161*.

[43] C. Luo, X. Liu, X. Chen, and J. Luo, "Multi-agent fault-tolerant reinforcement learning with noisy environments," in *Proc. IEEE 26th Int. Conf. Parallel Distrib. Syst.*, 2020, pp. 164–171.

[44] O. Kilinc and G. Montana, "Multi-agent deep reinforcement learning with extremely noisy observations," 2018, *arXiv:1812.00922*.

[45] F. Wu, S. Zilberstein, and X. Chen, "Online planning for multi-agent systems with bounded communication," *Artif. Intell.*, vol. 175, no. 2, pp. 487–511, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370210001578

[46] M. O. Karabag, C. Neary, and U. Topcu, "Planning not to talk: Multi-agent systems that are robust to communication loss," in *Proc. 21st Int. Conf. Auton. Agents Multiagent Syst.*, Virtual Event, New Zealand, 2022, pp. 705–713.

[47] B. Freed, G. Sartoretti, J. Hu, and H. Choset, "Communication learning via backpropagation in discrete channels with unknown noise," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 7160–7168. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/6205

[48] A. Mostaani, O. Simeone, S. Chatzinotas, and B. Ottersten, "Learning-based physical layer communications for multi-agent collaboration," in *Proc. IEEE 30th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.*, 2019, pp. 1–6.

[49] T.-Y. Tung, S. Kobus, J. R. Pujol, and D. Gunduz, "Effective communications: A joint learning and communication framework for multi-agent reinforcement learning over noisy channels," 2021. [Online]. Available: https://arxiv.org/abs/2101.10369

[50] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, pp. 99–134, 1998.

[51] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[52] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2015, *arXiv:1506.02438*.

[53] J. Zhou et al., "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[54] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[55] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Representations*, Vancouver, BC, Canada, Apr. 30–May 3, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[56] Z. Wang and M. Gombolay, "Learning scheduling policies for multi-robot coordination with graph attention networks," *IEEE Robot. Automat. Lett.*, vol. 5, no. 3, pp. 4509–4516, Jul. 2020.

[57] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 3431–3440, 2014.

[58] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with Gumbel-Softmax," in *Proc. Int. Conf. Learn. Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=rkE3y85ee

[59] N. Balachandar, J. Dieter, and G. S. Ramachandran, "Collaboration of ai agents via cooperative multi-agent deep reinforcement learning," 2019, *arXiv:1907.00327*.

[60] L. Han et al., "Grid-wise control for multi-agent reinforcement learning in video game ai," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2576–2585.

[61] D. Schwab, Y. Zhu, and M. Veloso, "Tensor action spaces for multi-agent robot transfer learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5380–5386.

[62] L. Pimentel, R. Paleja, Z. Wang, E. Seraj, J. Pagan, and M. Gombolay, "Scaling multi-agent reinforcement learning via state upsampling," Sandia Nat. Lab.(SNL-NM), Albuquerque, NM, USA, Tech. Rep. SAND2022-8384C; 707488, 2022. [Online]. Available: https://www.osti.gov/biblio/2003731

[63] M. Samvelyan et al., "The starcraft multi-agent challenge," in *Proc. 18th Int. Conf. Auton. Agents Multiagent Syst.*, Montreal QC, Canada, 2019, pp. 2186–2188.

[64] A. Vale, R. Ventura, J. Corisco, N. Catarino, N. Veiga, and S. Sargento, "Heterogeneous drone fleet for radiological inspection," in *Unmanned Aerial Vehicles Applications: Challenges and Trends*. Berlin, Germany: Springer, 2023, pp. 127–168.

[65] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of PPO in cooperative, multi-agent games," in *Proc. 36th Conf. Neural Inf. Process. Syst. Datasets Benchmarks Track*, 2022. [Online]. Available: https://openreview.net/forum?id=YVXaxB6L2Pl

[66] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 6382–6393.

[67] Proakis, *Digital Communications*. 5th Ed. New York, NY, USA: McGraw Hill, 2007.

[68] I. Otung, *Noise in Communication Systems*, 2021, pp. 431–472.

[69] J. A. Connelly, *Low-Noise Electronic System Design*, 1st ed. Hoboken, New Jersey, USA: Wiley, 1993.

[70] E. Seraj, X. Wu, and M. Gombolay, "Firecommander: An interactive, probabilistic multi-agent environment for joint perception-action tasks," 2020, *arXiv:2011.00165*.

[71] M. A. Finney, *FARSITE, Fire Area Simulator–Model Development and Evaluation*, Fort Collins, CO, USA: US Department of Agriculture, Forest Service, Rocky Mountain Research Station, 1998.

[72] E. Seraj et al., "Learning efficient diverse communication for cooperative heterogeneous teaming," in *Proc. 21st Int. Conf. Auton. agents multiagent Syst.*, 2022, pp. 1173–1182.

[73] S. Wilson et al., "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Syst.*, vol. 40, no. 1, pp. 26–44, Feb. 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:210695356

Esmaeil Seraj (Student Member, IEEE) received the Ph.D. degree in ECE from Georgia Tech, Atlanta, GA, USA, in 2023.

His Ph.D. research lies in the intersection of controls and robot learning, multiagent reinforcement learning, cooperative teaming, and learning emergent communication. He is currently an Applied Scientist with the Amazon Robotics, where he conducts research in robotics perception and planning for scalable robot manipulation.

Dr. Seraj publication record includes numerous publications across various conferences in ML/AI and Robotics and a best student paper nomination at ACC 2020.

Rohan Paleja (Student Member, IEEE) received the B.S. and M.S. degrees in mechanical engineering from Rutgers University, New Brunswick, NJ, USA, in 2017 and 2018, respectively, and the Ph.D. degree in robotics from Georgia Tech, Atlanta, GA, USA, in 2023.

He is currently a Technical Staff Research with the AI Technology Group, MIT Lincoln Laboratory. His research lies in the intersection of robot learning, multiagent coordination, and human–robot interaction.

Dr. Paleja publication record includes abundant publications across various conferences in AI and Robotics and a CoRL best paper nomination in 2020.

Luis Pimentel received the B.S. degree in computer engineering with a Robotics minor in 2021 from the Georgia Institute of Technology, Atlanta, GA, USA, where he is currently working toward the M.S. degree in electrical and computer engineering.

His current research interests lie in multiagent reinforcement learning, particularly developing scalable and generalizable learning algorithms for coordinating multiagent systems.

Kin Man Lee received the B.S. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 2016, the M.S. degree in robotics in 2023 from the Georgia Institute of Technology, Atlanta, GA, USA, where he is currently working toward the Ph.D. degree in robotics.

He is particularly interested in developing methods for robots to learn agile motions that can actively adapt to human heterogeneity. His current research interests lie broadly at the intersection of machine learning, controls, and human–robot interaction.

Zheyuan Wang received the B.S. degree in information and communication engineering and the M.S. degree in electrical and communication engineering both from Shanghai Jiao Tong University, Shanghai, China, in 2013 and 2016, respectively, and the M.S. and Ph.D. degrees in electrical and computer engineering from Georgia Tech, Atlanta, GA, USA, in 2016 and 2022, respectively.

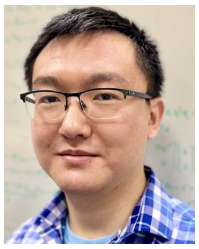
He is currently a Research Engineer with HAOMO.AI. His research lies in the intersection of multirobot coordination and stochastic resource optimization.

Daniel Martin received the bachelor's and master's degrees in computer science from the Georgia Institute of Technology, Atlanta, GA, USA, in 2021 and 2022, respectively.

He is currently working for Amazon Robotics, Boston, MA, USA. His research focused on robotics, primarily involving robot communication, robot learning, and learning from demonstration in sports applications, such as tennis and ping pong.

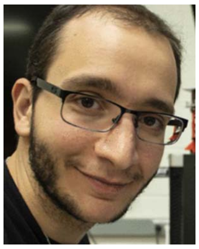
Matthew Sklar received the B.S. degree in computer science from Georgia Tech, Atlanta, GA, USA, in 2021, and the M.S. degree in computer science from the Georgia Institute of Technology, Atlanta, GA, USA, in 2022.

He is currently a Software Developer with Amazon. His current research interests are machine learning, intelligent planning systems, and multiagent coordination.



John Zhang (Graduate Student Member, IEEE) received the B.S. degree in mechanical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, and the M.S. degree in robotics from Carnegie Mellon University, Pittsburgh, PA, USA. He is currently working toward the Ph.D. degree in robotics with the Robotics Institute, Carnegie Mellon University.

His research interests include numerical optimization algorithms for modeling and decision-making of robotic systems.



Zahi Kakish received the B.S. degree in mechanical polymer engineering and the M.S. degree in mechanical engineering both from the University of Akron, Akron, OH, USA, in 2013 and 2015, respectively, and the Ph.D. degree in mechanical engineering from Arizona State University, Tempe, AZ, USA, in 2021.

He is currently a Senior Member of Technical Staff with Sandia National Laboratories in Albuquerque, NM, USA. His research interests include swarm intelligence, multiagent reinforcement learning, and robot autonomy in hazardous environments.

Matthew Gombolay (Member, IEEE) received the B.S. degree in mechanical engineering from Johns Hopkins University, Baltimore, MD, USA, in 2011, the S.M. degree in AeroAstro and the Ph.D. degree in autonomous systems from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2013 and 2017, respectively.

He is currently an Associate Professor of interactive computing with the Georgia Institute of Technology, Atlanta, GA, USA. Before starting as faculty, he worked with MIT's Lincoln Laboratory, transitioning his research to the U.S. Navy and earning an R & D 100 Award.

Dr. Gombolay is a DARPA Riser, a NASA Early Career Fellowship, and an NSF CAREER awardee.