Graph Neural Networks for Multi-Robot Active Information Acquisition

Mariliza Tzes*, Nikolaos Bousias*, Evangelos Chatzipantazis and George J. Pappas

Abstract—This paper addresses the Multi-Robot Active Information Acquisition (AIA) problem, where a team of mobile robots, communicating through an underlying graph, estimates a hidden state expressing a phenomenon of interest. Applications like target tracking, coverage and SLAM can be expressed in this framework. Existing approaches, though, are either not scalable, unable to handle dynamic phenomena or not robust to changes in the communication graph. To counter these shortcomings, we propose an Information-aware Graph Block Network (I-GBNet), an AIA adaptation of Graph Neural Networks, that aggregates information over the graph representation and provides sequential-decision making in a distributed manner. The I-GBNet, trained via imitation learning with a centralized sampling-based expert solver, exhibits permutation equivariance and time invariance, while harnessing the superior scalability, robustness and generalizability to previously unseen environments and robot configurations. Numerical simulations on significantly larger graphs and dimensionality of the hidden state and more complex environments than those seen in training validate the properties of the proposed architecture and its efficacy in the application of localization and tracking of dynamic targets.

I. INTRODUCTION

Over recent years, we have seen UAVs participating in the fighting of wildfires in California, robotic snakes assisting in localization of earthquake victims or UUVs deployed for underwater ocean mapping [1]. In such scenarios, autonomous robots need to sequentially make decisions that would allow them to better estimate the extent of the fire and decide upon extinguishing locations. The process of designing efficient paths to actively estimate a hidden state that expresses such a phenomenon of interest by utilizing measurement readings of on-board sensors is known in the literature as Active Information Acquisition (AIA). An exhaustive list of applications can be formulated in the AIA setup, from environmental monitoring, search & rescue, surveillance and coverage, target tracking and localization and active-SLAM [2]–[6].

A plethora of existing works [7]–[17] are proposed that solve the AIA problem. Myopic approaches [7]–[9] rely on computing controllers that incur the maximum immediate decrease of an uncertainty measure while search-based non-myopic schemes [10]–[12] solve the problem by pruning the exploration process. The latter return suboptimal solutions while their decentralized counterparts rely on coordinate descent, making them computationally intractable as the planning horizon and/or the number of robots increases. Non-myopic sampling-based approaches [13]–[17] gained popularity due to their ability to compute informative paths fast. All these approaches suffer from the following limitations: 1) they do not scale, e.g. for more than dozens of robots, 2) are not robust to changes of the problem’s parameters, e.g. number of agents or changes to the connectivity of robots and/or 3) do not address dynamic phenomena of interest.

Graph Neural Networks (GNNs) are information processing architectures for signals on graphs. Thanks to their distributed structure, GNNs can easily be adapted to multi-robot applications where the nodes and edges represent the robots and communication links respectively. They follow a neighborhood aggregation scheme and are therefore inherently suitable for distributed implementations. GNNs are also promising for their transferability across different graphs and can be designed such that they generalize to previously unseen multi-robot scenarios. Their use has the potential to radically increase (a) robustness to changes in the topology of the graph and (b) scalability with respect to the number of robots.

In this paper, we propose a novel architecture based on GNNs that solves the AIA problem in a distributed manner and addresses the aforementioned challenges. Specifically, we translate the multi-robot information gathering problem to a graph representation and formulate it as a sequential decision-making problem, where an Information-aware Graph Block Network (I-GBNet) learns to derive control actions that drive the robots to actively estimate the state of the phenomenon of interest. At every timestep, each robot collects positions and estimates from its neighbors and feeds them along with its own attributes to the I-GBNet that produces the next control inputs to be applied; see also Fig.1. To train our network, we use imitation learning where the
expert is a centralized sampling-based algorithm.

**Literature Review:** An exhaustive literature has been developed that addresses the AIA problem. A centralized sampling-based approach is introduced in [15] that explores the joint physical and information space of the robots. In [16] the authors propose a decentralized Monte Carlo Tree Search [18] method that samples the individual action space of each robot and then creates a sparse approximation of the joint action space. In [17], the authors design a distributed sampling-based approach where robots build their own trees and share with their neighbors information of randomly sampled nodes of their trees. The authors in [19] propose a more scalable approach by tessellating the environment into Voronoi diagrams, but an all-to-all communication and static hidden state is assumed. Data-driven methods [20]–[23] have also been proposed to allow for online implementations. Prior work in data-driven information acquisition utilizes imitation learning via a clairvoyant oracle [21, 24]. In [22] the authors develop a deep reinforcement learning technique that combines tree search with an offline-learned neural network, however it is only applicable to a single robot.

To the best of our knowledge, GNNs have never been used before to frame an AIA setting. Noticeably, they recently started gaining popularity in multi-robot applications [25]–[32]. In [26] GNNs are employed for path planning where an attention mechanism prioritizes meaningful information and the authors in [28], [29] propose GNNs for training control policies for multi-robot coverage. GNNs have also been exploited for SLAM [30], [32] and target tracking [31] where common in these works is the absence of notion of uncertainty over the robots’ observations. Compared to these works, our scheme provides directions on to where the robots should take measurements on-the-fly to reduce their underlying uncertainty.

**Contributions:** This paper proposes the first method for Active Information Acquisition using Graph Neural Networks. We designed a network that is scalable with respect to the number of robots and dimensionality of the hidden state to be estimated and generalizes to previously unseen multi-robot configurations. Our method is robust to communication failures and can be applied for time-varying communication graphs and dynamic hidden states. Additionally, we provide a variety of quantitative and qualitative simulations that illustrate the efficacy of our proposed architecture for the example of target localization and tracking.

## II. PROBLEM FORMULATION

Consider a homogeneous team of $N$ mobile robots that reside in a complex environment $T \subset \mathbb{R}^2$. The environment is cluttered with a set of arbitrarily shaped obstacles $O \subset T$, thus forming the obstacle-free area $T_{\text{free}} := T \setminus O$. The robots are governed by the robot dynamics $p_{i,t+1} = f(p_{i,t}, u_{i,t})$, $\forall i \in \{1, \ldots, N\}$ where $p_{i,t} \in T_{\text{free}}$ and $u_{i,t} \in \mathcal{U}$ denote the state of robot $i$ at time $t$ and control action selected from a finite set of admissible control actions, $\mathcal{U}$. The robots are tasked with collaboratively estimating a hidden state $x_i \in \mathbb{R}^d$ with hidden state dynamics $x_{i,t+1} = Ax_i + w_{i,t}$, where $w_{i,t} \sim N(0, Q_i)$ is the additive noise and $Q_i$ is the noise covariance matrix. The robots are equipped with sensors capable of collecting noisy measurements of the hidden state as per the observation model

$$y_{i,t} = M(p_{i,t}, x_i) + v_{i,t}, \quad \nu_{i,t} \sim N(0, R_i)$$

where $y_{i,t}$ is the measurement of robot $i$ at time $t$. We assume that the transition matrices $\{A, M\}$ and the noise covariance matrices $\{Q_i, R_i\}$ are known, similar to [17] and the robots perfectly localize themselves in a global frame. Hereafter, when subscript $i$ is omitted we denote compactly attributes of all robots.

### A. Multi-Robot Active Information Acquisition

The AIA problem requires the robots to collaboratively estimate the hidden state $x_i$. To do so, the robots maintain a prior Gaussian distribution for the hidden state, $x_0 \sim N(\mu_0, \Sigma_0)$. Given the prior distribution and measurements received up to time $t$, $y_{0:t}$, the à-posteriori distribution is computed i.e., $x_t \sim N(\mu_t(y_{0:t}), \Sigma_t(y_{0,t}))$ where $\mu_t(y_{0:t})$ and $\Sigma_t(y_{0,t})$ denote the à-posteriori mean and covariance matrix respectively. An indicative function of the overall quality of the measurements up to time $t$ in estimating $x_i$ is the determinant of the à-posteriori covariance matrix, i.e $\det \Sigma_t(y_{0:t})$ [33]. Alternative information measures could be used such as the trace or the maximum eigenvalue of the covariance. Given initial robots’ positions $p_0$ and prior distribution $\hat{x}_0$, the goal is to compute a planning horizon $F$ and a sequence of control inputs $u_{0:F}$, which solves the following deterministic optimal control problem [15]:

$$\min_{F, u_{0:F}} J(F, u_{0:F}) = \sum_{t=0}^{F} \det \Sigma_{t+1}$$

where the objective (2a) captures the cumulative uncertainty in the estimation of $x_t$ after fusing information collected by all robots from $t = 0$ up to time $F$ and the constraint in (2b) requires that the final uncertainty of the state $x_F$ is less than a user-specified threshold $\epsilon$. In (2c) we require obstacle free paths and (2d) describes the robot dynamics. Finally in (2e), $\rho(\cdot)$ stands for the Kalman Filter update rule that computes the à-posteriori distribution. A Centralized Sampling-Based (C-SB) method that solves Problem (2) is introduced in [15, Section III] that is probabilistically complete and asymptotically optimal.

**Remark 1:** The optimization problem defined in (2) resulted from a stochastic optimal control problem defined in [34] where a separation principle is applied due to the linear Gaussian assumptions.

### B. Problem Statement

Given initial positions of the robots $p_0$, capable of taking measurements as per the observation model (1), communicating via an underlying communication network and prior
estimates \( \hat{x}_0 \), we design a GNN architecture that (i) solves (2) in a distributed manner (ii) is trained using imitation learning to mimic the C-SB expert and (iii) derives sequential control actions \( u_t \) that allow the robots to actively decrease their uncertainty over \( x_t \).

### III. Graph Neural Networks for AIA

To solve the Multi-Robot Active Information Acquisition problem described in Section II in a distributed way with GNNs, we develop a suitable graph representation of the problem. At every timestep the graph is given as an input to the network that consists of an Information-aware Graph Block Network (I-GBNet) and a Multi-Layer Perceptron (MLP). I-GBNet updates the node attributes of the incoming graph and the updated attributes are fed to the MLP that derives the control actions for all robots.

#### A. Graph Representation of AIA problem

In the distributed setting, we assume that each robot maintains a local estimate over the hidden state expressed via its a-posteriori Gaussian distribution \( \hat{x}_{i,t} \sim \mathcal{N}(\mu_{i,t}, \Sigma_{i,t}) \). For simplicity reasons, we define \( Z_{i,t} := M(p_{i,t})^T \Sigma_{i,t}^{-1} M(p_{i,t}) \) and the information matrix \( \Omega_{i,t} := \Sigma_{i,t}^{-1} \).

Let the robots communicate through an underlying communication network modeled as a directed communication graph \( G_{c,t} = (\mathcal{V}_c, \mathcal{E}_c, \mathcal{T}_c) \). The set of vertices \( \mathcal{V}_c := \{1, \ldots, N\} \) are indexed by the robots and an edge \( (i,j) \in \mathcal{E}_c \subseteq \mathcal{V}_c \times \mathcal{V}_c \) defines a direct communication between robots \( i \) and \( j \). Robot \( i \) can exchange information with the set of its neighbors \( \mathcal{N}_{i,t} \). Hereafter we define \( \mathcal{N}_{i,t} := \mathcal{N}_{i,t} \cup \{i\} \).

Given the aforementioned communication graph \( G_{c,t} \), let the state of the problem at timestep \( t \) be described by the following homogeneous, weighted, directed network graph:

\[
\mathcal{G}_t = \{ \mathcal{V}_t, \mathcal{E}_t, \mathcal{T}_t; \mathcal{G}_t \}
\]

where \( \mathcal{V}_t = \{ \mathcal{V}_{i,t} \}_{i=1}^N \) is the set of node attributes of cardinality \( |\mathcal{V}_t| = N \), indexed by the robots \( i \in \mathcal{V}_c \), \( \mathcal{E}_t \subseteq \mathcal{V}_t \times \mathcal{V}_t \) is the set of edges, \( \mathcal{W}_t \subseteq \mathcal{R}^{N \times N} \) is the weight matrix and \( \mathcal{G}_t \) represents the global attributes of the graph (e.g. environment \( T \)). Each node (robot) \( i \) contains the following node attribute:

\[
\mathcal{V}_{i,t} = \{ p_{i,t}, Z_{i,t}, \mu_{i,t}, \Omega_{i,t} \}
\]

The weight matrix quantifies the importance of information being propagated from robot \( i \) to \( j \) and is defined as \( W_{t}|_{ij} = \kappa_{ij} I_{(i,j) \in \mathcal{E}_t} \), where \( \kappa_{ij} > 0 \) and \( \sum_{j \in \mathcal{N}_{i,t}} \kappa_{ij} = 1 \).

#### B. Information-aware Graph Block Network

The core of our GNN is I-GBNet \( \Phi \), illustrated in Fig.2 and inspired by [35]. I-GBNet receives a graph signal \( \mathcal{G}_t \) and updates the node attributes by propagating information through the graph, while keeping the graph topology unchanged, i.e. \( \mathcal{G}_t' = \Phi(\mathcal{G}_t) \). For the design of I-GBNet, we finely discretize the environment \( T \subset \mathcal{R}^2 \) and represent it via an occupancy grid-map \( \mathcal{T} \) of size \( \mathcal{T}_{u} \times \mathcal{T}_{d} \). The network \( \Phi \) comprises of three parts i) two aggregation functions \( \rho^\Omega(\cdot), \rho^T(\cdot) \), (ii) a heatmap projection function \( \mathcal{H} \) and (iii) a node-update function \( \phi^\nu \).

**Aggregation:** We assume that at timestep \( t \), the robot \( i \) receives information of the form \( \{ \mathcal{G}_t(j), \mathcal{I}(p_{j,t}) \}, \forall j \in \mathcal{N}_{i,t} \), where \( \mathcal{I}(\cdot) \) is a function that transforms the robot’s position into a binary robot-grid map. Let \( \Omega_{\mathcal{N}_{i,t},t} := \{ \mathcal{G}_t(j), \forall j \in \mathcal{N}_{i,t} \} \) and \( \mathcal{I} = \{ \mathcal{I}(p_{j,t}), \forall j \in \mathcal{N}_{i,t} \} \). Inspired by the active information problem setup and constraint (2e), the aggregation function \( \rho^\Omega \) is defined as the Distributed Kalman Filter (DKF), proposed in [33], where robot \( i \) updates its a-posteriori covariance matrix as follows:

\[
\hat{\Omega}_{i,t} := \rho^\Omega (\mathcal{G}_t; \mathcal{I}; \mathcal{W}_t) = \sum_{j \in \mathcal{N}_{i,t}} \kappa_{ij} \mathcal{G}_t + \mathcal{Z}_{i,t}
\]

Each robot, also, aggregates the positions of its neighbours into a neighbors-grid map as:

\[
\hat{p}_{i,t} := \rho^T (\mathcal{I}(\mathcal{T}_u \times \mathcal{T}_d); \mathcal{W}_t) = \sum_{j \in \mathcal{N}_{i,t}} \mathcal{I}(p_{j,t}) - \mathcal{I}(p_{i,t})
\]

The selected aggregation functions are invariant to permutations of their inputs by construction.

**Heatmap Projection:** Once the robot computes the aggregated covariance matrix \( \hat{\Omega}_{i,t} \), it projects it into the occupancy grid map \( \mathcal{T} \) via the projection function \( \mathcal{H} : \{ \mathcal{G}_t, \mathcal{I}, \mathcal{W}_t \} \rightarrow \mathcal{H}_T \), where \( \mathcal{H}_T \) is the resulting grid heatmap that represents the magnitude of uncertainty in \( \mathcal{T} \). In the case of multi-target localization, the projection module would produce a combination of the posterior Gaussian heatmaps.

**Node-Update:** Given the aggregated neighbors-grid map \( \hat{p}_{i,t} \), the robot-grid map \( \mathcal{I}(p_{i,t}) \), the heatmap-grid \( \mathcal{H}_T \) and the occupancy-grid map \( \mathcal{T} \), the node-update function \( \phi^\nu \) produces the control actions

\[
u_{i,t} := \phi^\nu (\hat{p}_{i,t}, \mathcal{I}(p_{i,t}), H_{i,t}, T)
\]

hence the updated node attributes are \( \mathcal{V}'_{i,t} = \{ u_{i,t} \} \). The node-update function \( \phi^\nu \) is parametrized by a Convolutional Neural Network with learnable parameters \( \theta_{\text{CNN}} \) composed with an Action-MLP of parameters \( \theta_{\text{MLP}} \). The output of the Action MLP produces a categorical distribution over the set of admissible control inputs \( \mathcal{U} \) from which the most probable action \( u_{i,t} \) is selected.

#### C. Properties of Graph Block Network

The graph topology is expected to vary over time owing to the mobility of the agents. Therefore, the Graph Block is required to perform consistently for all permutations of \( \mathcal{N}_{i,t} \subset \mathcal{V}_c \), i.e. swapped order of robot indices, and regardless of the time shift. The permutation equivariance property for the aforementioned Graph Block network, directly limits the required sample complexity for training the network.

**Proposition 1:** (Permutation equivariance of Graph Block) Given the aforementioned graph \( \mathcal{G}_t \), for any permutation \( \pi : [n] \rightarrow [n] \) of \( \mathcal{V}_c \), with corresponding permutation matrix \( \mathcal{P}_\pi \), and the Graph Block defined in subsection III-B, it holds that \( \mathcal{P}_\pi (\Phi(\mathcal{G}_t)) = \pi(\Phi(\mathcal{G}_t)) \).

**Proof:** For the purposes of the proof, we denote \( \hat{\Omega}_t := \oplus_{i \in [N]} \hat{\Omega}_t^i, \Omega_t := \oplus_{i \in [N]} \Omega_t^i, Z_t := \oplus_{i \in [N]} Z_t^i, \hat{p}_t := \oplus_{i \in [N]} \hat{p}_t^i \), and \( \mathcal{I}_t := \oplus_{i \in [N]} \mathcal{I}(p_{i,t}) \), where the operator \( \oplus \)
denotes a concatenation of matrices along the first dimension. Let $A \in \mathbb{R}^{N \times N}$ denote the adjacency matrix of the graph $G_i$. The per-agent aggregation process, described by (5) and (6), can be reformulated as

$$\hat{\Omega}_t = \left[ \rho^I_{\Omega}(\hat{\Omega}_{N_x(t), t}, Z_{I}, t; W_{I}) \right]^{T} \cdots \left[ \rho^I_{\Omega}(\hat{\Omega}_{N_x(t), t}, Z_{I}, t; W_{I}) \right]^{T} = (\nu_{I} \circ \delta_{d_{1}}) \Omega_{t} + Z_{I}$$

$$\hat{P}_t = \left[ \rho^P(\hat{P}_{N_x(t), t})^{T} \cdots \rho^P(\hat{P}_{N_x(t), t})^{T} \right]^{T} = ((A_{I} - I_{N}) \circ I_{T_H}) I_{I}$$

Then, the permutation of the aggregated node attributes is the aggregation of the permuted node attributes as

$$\left[ \rho^I_{\Omega}(\hat{\Omega}_{N_x(1), t}, t; Z_{I}(1); \pi(W_{I})) \right] \cdots \left[ \rho^I_{\Omega}(\hat{\Omega}_{N_x(N), t}, t; Z_{I}(N); \pi(W_{I})) \right] = (P_{\pi}W_{1} \circ \delta_{d_{1}}) \Omega_{t} + (P_{\pi} \circ \delta_{d_{1}}) Z_{I}$$

and similarly,

$$\left[ \rho^P(\hat{P}_{N_x(1), t})^{T} \cdots \rho^P(\hat{P}_{N_x(N), t})^{T} \right]^{T} = (P_{\pi}(A_{I} - I_{N}) \circ I_{T_H}) (P_{\pi} \circ I_{T_H}) I_{I} = \pi(\hat{P}_t)$$

Since the update function is applied on a per-node basis on the aggregated permuted node attributes

$$V_{\pi(1), t} = \left[ \phi^\nu(\hat{P}_{\pi(1), t}, I_{\pi}(\hat{P}_{\pi(1), t}), H_{\pi(1), t}; T) \right]$$

$$\vdots$$

$$V_{\pi(N), t} = \left[ \phi^\nu(\hat{P}_{\pi(N), t}, I_{\pi}(\hat{P}_{\pi(N), t}), H_{\pi(N), t}; T) \right]$$

$$= (P_{\pi} \circ I_{u}) \left[ \phi^\nu(\hat{P}_{\pi(1), t}, I_{\pi}(\hat{P}_{\pi(1), t}), H_{\pi(1), t}; T) \right]$$

Thus, as the topology of the graph remains unchanged, we conclude $\Phi(\pi(G_{i})) = \pi(\Phi(G_{i}))$.

**Proposition 2:** (Time invariance of Graph Block)

For $t_1, t_2 \in \mathbb{R}^+$, $t_1 \neq t_2$, given $G_{t_1} = \{V_{c}, V_{t_1}, \mathcal{E}_{t_1}, \mathcal{W}_{t_1}; g\}$, $G_{t_2} = \{V_{c}, V_{t_2}, \mathcal{E}_{t_2}, \mathcal{W}_{t_2}; g\}$ where $V_{t_1} = V_{t_2}, \mathcal{E}_{t_1} = \mathcal{E}_{t_2}, \mathcal{W}_{t_1} = \mathcal{W}_{t_2}$, and the Graph Block $\Phi$ defined in subsection III-B, it holds that $\Phi(G_{t_1}) = \Phi(G_{t_2})$.

**Proof:** This proposition is satisfied naturally owing to the imitation learning method explained in the following Section and the fact that none of the components in $\Phi$ is an explicit function of time. The Graph Block network is trained to enable each robot to predict actions consistently and independently of $t$, given a graph $G_{i}$, imitating expert decision making.

---

**IV. TRAINING AND ARCHITECTURE**

**A. Dataset and Training**

For the dataset creation, we run the C-SB expert for 1500 randomly generated $20 \times 20$ environments cluttered with a fixed number of obstacles, 10 robots with first-order dynamics and sensing radius of 2$m$, static hidden state $x \in \mathbb{R}^{20}$ and $U = \{left, up, right, down\}$ by 0.5$m$. The uncertainty threshold $\epsilon$ was fixed to $1e^{-1}$. For each instantiation we run entire episodes and randomly select 10 timesteps out of each to collect 15000 sets $\{(p^k, z^k, m^k, \Omega^k, t^k), u^k\}$. The latter sets are then transformed via the graph representation process (Section III-A) and result in our dataset $D = \{G^k, u^k\}$. Given dataset $D$ and the learnable parameters of the I-GBNet, $\theta_{CNN}$ and $\theta_{MLP}$, we minimize the cross-entropy loss:

$$\min_{\theta_{CNN}, \theta_{MLP}} \left[ \sum_{(G^k, u^k) \in D} L_{ce}(\Phi(G^k), u^k) \right]$$

For training, the dataset is divided into a training set (90%) and a validation set (10%). The training is conducted with the Adam optimizer and the learning rate is scheduled to decay from $10^{-4}$ to $10^{-6}$ within 20 epochs with cosine annealing. Finally, we trained with batch size 16 and weight decay equal to $10^{-5}$. **B. Network Architecture**

The architecture features a single-layer Graph Block, described in Section III-B, to provide the back bone of the distributed coordination scheme. We parametrize the learnable, node update function $\phi^\nu(\cdot)$, illustrated in Fig.2 as follows: Initially, a ResNet module implements feature extraction from the 4 channels and comprises of 3 sequential 2-layer (Conv2d-BatchNorm2d-ReLU) Residual Blocks with skip connections, followed by an adaptive average-pooling...
module. The latter provides the network with the flexibility to process environments of variable size by projecting to a fixed representation. The final layer is an MLP with parameters \( \theta_{\text{MLP}} \), that translates this representation into probabilities per control action.

V. EXPERIMENTS

In this Section, we provide simulations to demonstrate the efficacy of our method. All case studies have been implemented using Python3 on a computer with a 3.2 GHz, 8-Core, Intel Core i7-800H CPU, 64GB RAM and an Nvidia RTX 3080Ti GPU, 16GB RAM. The experiments were conducted assuming that the robots can take noisy measurements with a camera that provides the \( xy \)-coordinates of any visible object within its sensing range, expressed in the robot’s frame, where the sensing radius is selected to be equal to \( 2m \) and the noise covariance matrix \( \mathbf{R}_{t} = (0.05 \circ x_t)^2 \mathbf{I} \). We assume first-order dynamics robots that select any action from the set \( \mathcal{U} = \{ \text{left, right, up, down} \} \) by 0.5m and the DKF parameters are chosen as \( \kappa_{ii} = 0.75 \) and \( \kappa_{ij} = 0.25/|\mathcal{U}| \). We test our algorithm on the applications of target localization and tracking, while the hidden state \( x_t \) collects the positions of all targets at time \( t \), i.e. \( x_t = [x_{1,t}^T, \ldots, x_{M,t}^T]^T \), where \( x_{k,t} \) is the position of target \( k \) at time \( t \) and \( M \) is the total number of targets. In this setting, we require to drop the uncertainty for all targets by at least one robot, i.e. \( \det \Sigma_k^T \leq \epsilon \). To achieve collision avoidance when an invalid action is given, the robot randomly selects a collision-free action from \( \mathcal{U} \), otherwise stays idle. A more sophisticated collision avoidance method [36]–[38] could be used instead. Unless stated differently, we assume a fully connected graph. It is important to stress that for all the experiments below we use the same network, trained as described in IV-A and without any fine-tuning per task.

A. Scalability and Generalization

We are interested in examining how our network, trained on \( 20m \times 20m \) (200 m x 200 occupancy grid maps) with \( N = 10 \) robots and a hidden state \( x_t \in \mathbb{R}^{20} \), would perform on previously unseen \( 20m \times 20m \) environments and varying number of robots \( N \) and targets \( M \). For this experiment, we run the Expert (C-SB) for each of the following instances \( N \times M = [10 \times 10, 20 \times 20, 30 \times 30, 40 \times 40, 50 \times 50, 60 \times 60, 80 \times 80] \) for \( n = 100 \) random initializations (obstacles, robots and target positions) and compare our GNN approach with two baselines (a) a Random-Walker and (b) a Distributed Sampling-Based (Dec-SB) AIA algorithm [17]. We evaluate the algorithms similar to [25] based on the following metrics:

1) Flowtime Increase \( \delta_{FT} = (F - F^*)/F^* \), expresses the percentage change between the returned planning horizon \( F \) and the expert’s horizon \( F^* \). For each of the configurations, we collect \( n \) of the \( \delta_{FT} \)'s and return as the flowtime increase of the configuration the mean.

2) Success Rate = \( n_{\text{success}}/n \), where \( n_{\text{success}} \) is the number of successful cases out of the \( n \) cases of the expert. A case is considered successful when a solution is acquired with a planning horizon \( F \leq 3F^* \).

From Fig. 3a, we can see that the success rate of I-GBNNet is equally well and occasionally better than the Dec-SB algorithm and always outperforms the Random Walker’s. The Random-Walker performs better on more condensed environments (robots randomly initialized closer to targets) while the Dec-SB could not frequently return solutions with planning horizons less than \( 3F^* \). Even in the case of condensed environments where we expect smaller planning horizons returned by the expert, our method manages to complete the tasks within the horizon bounds. In Fig. 3b, the mean of the Flowtime Increases of our method is systematically smaller than those of the baselines. We conclude that our network, trained only on environments of 10 targets and 10 robots is able to learn policies that generalize to novel environments with much larger graphs and hidden states. Moreover, mimicking a centralized expert is able to learn a distributed policy that is usually more effective than hand-designed distributed policies for the same problem.

B. Target Localization and Tracking

In this experiment, we are interested in validating our method qualitatively where the uncertainty threshold is set \( \epsilon = 1.8e^{-2} \) for each of the targets. In the first scenario, three robots reside on a randomly-generated \( 20m \times 20m \) environment cluttered with a dense number of obstacles and are tasked to localize 5 static landmarks, while avoiding the
In this paper, we reduce the multi-robot Active Information Acquisition problem into a learning framework over graphs and introduce I-GBNet, an AIA-inspired GNN, that offers distributed decision-making, imitating a centralized expert. In contrast to other works, our method can deal with dynamic phenomena and is robust to changes in the communication graph. Experiments demonstrated that despite being trained on small graphs, our network successfully generalizes to previously unseen environments/robot configurations and scales well. In the future, we shall investigate on more applications of Active Information Acquisition, like occupancy mapping and we shall explore the applicability of attention layers to further encourage scalability and time-efficiency.

VI. Conclusions

In this paper, we reduce the multi-robot Active Information Acquisition problem into a learning framework over graphs and introduce I-GBNet, an AIA-inspired GNN, that offers distributed decision-making, imitating a centralized expert. In contrast to other works, our method can deal with dynamic phenomena and is robust to changes in the communication graph. Experiments demonstrated that despite being trained on small graphs, our network successfully generalizes to previously unseen environments/robot configurations and scales well. In the future, we shall investigate on more applications of Active Information Acquisition, like occupancy mapping and we shall explore the applicability of attention layers to further encourage scalability and time-efficiency.
REFERENCES


