Decentralized Global Connectivity Maintenance for Multi-Robot Navigation: A Reinforcement Learning Approach

Minghao Li, Yingrui Jie, Yang Kong, Hui Cheng

Abstract—The problem of multi-robot navigation of connectivity maintenance is challenging in multi-robot applications. This work investigates how to navigate a multi-robot team in unknown environments while maintaining connectivity. We propose a reinforcement learning (RL) approach to develop a decentralized policy, which is shared among multiple robots. Given range sensor measurements and the positions of other robots, the policy aims to generate control commands for navigation and preserve the global connectivity of the robot team. We incorporate connectivity concerns into the RL framework as constraints and introduce behavior cloning to reduce the exploration complexity of policy optimization. The policy is optimized with all transition data collected by multiple robots in random simulated scenarios. We validate the effectiveness of the proposed approach by comparing different combinations of connectivity constraints and behavior cloning. We also show that our policy can generalize to unseen scenarios in both simulation and holonomic robots experiments.

I. INTRODUCTION

Multi-robot navigation is one of the significant problems for multi-robot systems and practical to cooperative tasks such as search and rescue, formation control, and area coverage. As each robot has limited communication and sensing ranges, cooperative behaviors among robots rely on the connectivity of the communication network. Therefore, maintaining connectivity over time is important for multi-robot navigation.

The problem of connectivity maintenance for multi-robot systems has been widely studied, and various connectivity strategies [1]–[4] have been proposed. Most of these works consider global connectivity because of its flexibility as it allows robots to break communication links as long as the communication graph is connected over time. This is guaranteed by preserving the second-smallest eigenvalue of the underlying graph Laplacian of robot networks to be positive, which is well-known as algebraic connectivity [5]. As these works attempt to complete various tasks, some of them validate their strategies in obstacle-free environments for formation control [4], visiting multiple waypoints [6] and area coverage [7]. Potential field methods are widely studied to combine connectivity maintenance with system objectives and obstacle avoidance. Williams et al. [3] select a dispersive potential with collision avoidance to validate the designed connectivity controller at the presence of disc-obstacles. Li et al. [1] have proposed a potential-based method towards circle obstacles avoidance with connectivity maintenance. Sabattini et al. [2] evaluate their gradient-based controller in rendezvous applications. They use a collision avoidance strategy which is not mentioned in experiments as the controller is designed for general purposes. One of the major challenges of multi-robot navigation is the imperfect environment information due to limited sensing capabilities. Connectivity controllers may fail since they can be trapped at the presence of obstacles without the explicit design of navigation strategies. Moreover, simple assumptions on obstacle shapes have limitations in complex environments.

To navigate in complex environments, RL-based approaches are alternatives that have shown promising results in map-less navigation. Many of these works have developed end-to-end navigation policies which directly map raw sensor measurements to control commands without prior knowledge of the environment [8]–[10]. While few of them focus on connectivity of the robot team, Lin et al. [11] propose a RL framework that combines a constrained optimization module to enforce local connectivity constraints. However, local connectivity may lead to restrictive behaviors since each robot has to maintain communication links with all other robots in the team over time. Moreover, the effects of action correction from the optimization module are nontrivial in RL as it may break the Markov assumption [12, 13].

In this work, we focus on multi-robot navigation with connectivity maintenance at the presence of unknown obstacles. We consider global connectivity to allow flexible changes of communication graphs and incorporate it as constraints into RL methods. Although the connectivity-constrained multi-robot navigation problem can be formulated as a Constrained Partial Observable Markov Decision Process (Constrained POMDP), existing constrained RL methods are infeasible due to sample inefficiency and the inherent difficulty of this multi-objective problem (reaching target points and avoiding collisions while bounding connectivity constraints). To address this issue, we present a model-free constrained
RL approach that combines behavior cloning (BC) to make policy training tractable. By learning from demonstrations, the actions of robots are guided towards expert navigation behaviors and therefore improve efficiency in policy search. Following the centralized training decentralized execution scheme [14, 15], a shared policy is optimized by experiences of all agents during training. As a result, a decentralized policy is developed that takes information of other agents into account to maintain global connectivity and leverages range sensor data for collision avoidance.

The contributions of this work are:

- A constrained RL-based approach is proposed to navigate multiple robots in unknown environments while maintaining global connectivity in a decentralized manner.
- An effective way to facilitate policy training by combining BC and RL without breaking the connectivity constraint.
- The performance of the developed policy is empirically evaluated in simulation and real-world robotic platforms.

II. RELATED WORK

RL-based approaches have been intensively studied in the last few years and have shown great potentials in navigation with imperfect sensing in unknown complex environments. For single-robot navigation, Pfeiffer et al. [8] propose a sample efficient approach that uses expert demonstrations for policy pre-training and further improve the policy performance by RL. The experiments show that the policy outperforms their previous approach, where an end-to-end policy is trained by supervised learning only [16]. In particular, they treat collision avoidance as constraint to ensure safety and use pre-training to improve sample efficiency. In our approach, a connectivity-maintained expert is infeasible, and thus we impose connectivity constraints and provide navigation demonstrations in one-stage RL training without pre-training. Zhang et al. [17] aim to develop a flexible policy and propose a transfer method that can handle various range sensor dimensional configurations. Later, they propose a support-point approach to distinguish critical points from LiDAR range measurements and handle different LiDAR configurations without re-training.

In multi-robot settings, Chen et al. [18] propose an approach called CADRL that considers observable states of other robots and pedestrians instead of direct mapping from perception to motion commands. With a trained value function, the robots perform one-step lookahead to generate collision-free actions. Later, they extend CADRL and design a socially-aware reward function for learning a policy that incorporates social norms towards pedestrian-rich navigation. Everett et al. [19] augment this framework with Long short-term memory to improve performance on complex interactions with pedestrians. The parameter sharing scheme [20, 21] is popular for efficient multi-agent policy training and applicable to multi-robot navigation. Long et al. [9] train a multi-robot collision avoidance policy with a multi-agent extended variant of Proximal Policy Optimization (PPO) and adopt curriculum learning to facilitate training. This policy is shared among robots and navigates each robot to its desired goal from 2D laser measurements without communications in a decentralized manner. Lin et al. [10] aim to navigate the geometric center of a robot team to reach waypoints and develop an end-to-end policy shared among the robots that consider raw laser data inputs and position information of other robots. Han et al. [22] also consider observable states of other robots and dynamic obstacles with Gaussian noises. They propose a greedy target allocation method to efficiently navigate the robots to multiple targets.

III. APPROACH

A. Problem formulation

The multi-robot navigation problem can be formulated as a Partial Observable Markov Decision Process (POMDP). Formally, a POMDP is defined by tuple \( < S, A, O, \mathcal{P}, Z, r, \gamma > \) where \( S \) is the state space, \( A \) is the action space, \( O \) is the observation space, \( \mathcal{P}(s'|s, a) \) is the transition probabilities of taking action \( a \) at state \( s \), \( Z(o|s', a) \) is the observation distribution at the new state \( s' \), \( r(s, a) : S \times A \rightarrow \mathbb{R} \) is the reward function and \( \gamma \in (0, 1] \) is the discount factor.

Specifically, we consider a team of \( N \) robots navigate to a target zone \( g \) with radius \( R_g \). The robots are considered homogeneous and modeled as discs with the same radius \( R_c \) and communication range \( R_c \). At each timestep \( t \), each robot \( i \) obtains 2D laser scans \( o_i^{t} \) and its velocity \( v_i^{t} \). The relative goal position \( o_g^{t} \) and the positions of other robots in the team \( o_p^{t} = [p_1^{t}, \ldots, p_{i-1}^{t}, p_{i+1}^{t}, \ldots, p_N^{t}] \) are informed to the robot \( i \). Given the observation \( o_i^{t} = [o_i^{t}, o_v^{t}, o_g^{t}, o_p^{t}] \), the robot aims to obtain a parameterized policy \( \pi_\theta \) that directly maps \( o_i^{t} \) to control command \( a_i^{t} \). The policy \( \pi_\theta \) is shared among robots, and each robot is allowed to take different actions since they receive different observations.

The objective of RL is to find the optimal parameter \( \theta \), where the corresponding policy \( \pi_\theta \) maximizes the expected cumulative discounted rewards:

\[
J(\theta) = \mathbb{E} \left[ \sum_{t=0}^{T} \gamma^t r(s_t, a_t) \right]
\]
where $T$ is the horizon of an episode.

1) Observation space and action space: As mentioned above, the observation for each robot consists of four parts $[o_i, \alpha_i, o_g, o_p]$. The 2D laser scans data $o_i$ consists of 90 measurements from a 360° LiDAR, the velocity $\alpha_i$ is in Cartesian coordinate, and all of the positions (i.e., $o_g$ and $o_p$) are transformed to the local robot coordinate frame from global coordinates. In our case, the robots are omnidirectional and the control command is the velocity in Cartesian coordinate, i.e., $\alpha = [v_x, v_y]$. For practical purpose, we shrink the output of the policy $\pi_\theta$ to fit the maximum value of velocity, i.e., $\|a\| \leq v_{\text{max}}$.

2) Reward design: The robot team is supposed to get into the target zone without collisions. To achieve this, the reward function for each robot $i$ is designed to encourage approaching the target zone and avoiding collisions with other robots and obstacles $p_{\text{obs}}$:

$$r^i(s_t, a_t) = r^i_{\text{goal}} + r^i_{\text{coll}}$$

where $r^i_{\text{goal}}$ is the reward when the robot approaches the target, or all of the robots enter the target zone:

$$r^i_{\text{goal}} = \begin{cases} r_{\text{goal}} & \text{if } \text{dst}(j, t) < R_g - R_r \\
 w_g(\text{dt}(i, t-1) - \text{dst}(i, t)) & \text{otherwise} \end{cases}$$

where $\text{dst}(i, t)$ is the distance to goal for robot $i$ at timestep $t$ and $w_g > 0$. $r^i_{\text{coll}}$ is to penalize collisions among robots or obstacles:

$$r^i_{\text{coll}} = \begin{cases} r_{\text{coll}} & \text{if } \|p^i_j - p_{\text{obs}}\| < R_r \\
 \text{or } \|p^i_j - p^i_i\| < 2R_r & \forall j \in [1, N], j \neq i \\
 0 & \text{otherwise} \end{cases}$$

We empirically set $r_{\text{coll}} = -100$, $r_{\text{goal}} = 100$ and $w_g = 10$.

B. Global connectivity constraint

The approach for global connectivity maintenance of communication networks is based on the second smallest eigenvalue of the associated graph Laplacian, which is briefly described below and used to formulate a connectivity constraint later on.

Consider a multi-agent systems with positions denoted as $p_i \in \mathbb{R}^2, i \in \{1, 2, \ldots, N\}$. Each pair of agents $i$ and $j$ can communicate with each other within an identical limited distance $R_c$, i.e., $\|p_i - p_j\| \leq R_c$. The communication topology can be represented as an undirected graph $\mathcal{G} = (V, E)$ where each vertex $v_i \in V$ denotes the $i$-indexed agent and each edge $e_{ij} \in E$ denotes the communication availability between agents $i$ and $j$. The Laplacian matrix of graph $\mathcal{G}$ is defined as $L = D - A$ where $A \in \mathbb{R}^{N \times N}$ is the adjacency matrix and each element $a_{ij} = 1$ if $e_{ij} \in E$ and $a_{ij} = 0$ otherwise. $D \in \mathbb{R}^{N \times N}$ is the degree matrix, which is diagonal, i.e., $D = \text{diag}(\phi_1, \phi_2, \ldots, \phi_N)$ and the $i$-th element of the diagonal is $\phi_i = \sum_{j=1}^{N} a_{ij}$. The second smallest eigenvalue of the Laplacian matrix, denoted as $\lambda_2(\mathcal{G})$, describes algebraic connectivity of the graph $\mathcal{G}$ as $\lambda_2(\mathcal{G}) \geq 0$ always holds if $\mathcal{G}$ is connected [5].

With the RL formulation, the robot team learns to navigate to the target zone. To impose global connectivity to the team, we introduce a cost function $c(s_t, a_t): \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ that use the algebraic connectivity measurement:

$$c(s_t, a_t) = \mathbb{I}(\lambda_2(\mathcal{G}) < 0)$$

where $\mathbb{I}$ is the indicator function and $\lambda_2(\mathcal{G})$ is the algebraic connectivity of the graph $\mathcal{G}$, which is constructed by $[p_1^t, p_2^t, \ldots, p_N^t]$ at timestep $t$. Note that the robots can infer $\lambda_2(\mathcal{G})$ from positions of the teammates $o^t_i$. As global connectivity serves as a constraint, the multirobot navigation problem is extended to a constrained RL formulation. Specifically, constrained RL aims to optimize the policy while bounding expected discounted returns with respect to the cost function:

$$J_c(\theta) = \mathbb{E}\left[ \sum_{t=0}^{T} \gamma^t c(s_t, a_t) \right]$$

The Eq. (6) is formed in terms of the expected value of trajectories. Similar to [8], by setting the discount factor $\gamma_c$ close to 1, we are able to constrain the total number of constraint violations per trajectory to be less than the desired threshold $d$. In this work, we use Constrained Policy Optimization (CPO) [23] to solve this constrained RL problem:

$$\theta^* = \arg \max_{\theta} J(\theta)$$

s.t. $J_c(\theta) \leq d$

where $d$ is the upper bound. CPO adopts trust region methods for local policy search, which is introduced in Trust Region Policy Optimization (TRPO) [24]. TRPO uses a constraint on the Kullback-Leiber (KL) divergence of two stochastic policies between updates to stabilize policy training with monotonic improvement guarantees. Similarly, CPO replaces the original objective and constraint in Eq. (7) with surrogate functions in a trust region:

$$\theta^{k+1} = \arg \max_{\theta} \mathbb{E}_{s,a \sim d^{\pi_k}} \left[ \frac{\pi_\theta(s)}{\pi_k(s)} A^{\pi_k}(s, a) \right]$$

s.t. $J_c(\theta^k) + \mathbb{E}_{s,a \sim d^{\pi_k}} \left[ \frac{\pi_\theta(s)}{\pi_k(s)} A^{\pi_k}(s, a) \right] \leq d$

$$\tilde{D}_{KL}(\pi_\theta || \pi_{k}) \leq \eta$$

where $\tilde{D}_{KL}(\pi_\theta || \pi_{k})$ is the average KL divergence between two consecutive policies and $\eta > 0$ is the step size. Advantages functions $A^{\pi_k}(s, a), A^{\pi_k}(s, a)$ are introduced in Eq. (8) which can be estimated by function approximators $V_{\phi}, V_{\phi}$, with one-step TD error $A(s, a) = r(s, a) + \gamma V(s_{t+1}) - V(s_t)$. CPO approximately solves the surrogate optimization in Eq. (8). Due to approximation errors, CPO asymptotically bounds the constraint during training. One may refer to [23] for details.

C. Behavior cloning

In fact, the optimization problem in Eq. (7) regarding two objectives (i.e., Eq. (1) and Eq. (6)) is usually infeasible as it leads to conservative policy explorations in stochastic domains even without a trust region constraint. In our case,
connectivity constraint leads to in-place oscillations, and the robots cannot learn to navigate to the target.

To make this policy optimization tractable, we introduce the behavior cloning (BC) technique during RL training. In many robotic tasks such as manipulation, limited demonstration data can be collected, but an expert policy for all states is usually infeasible [25, 26]. However, for map-less navigation, recent successful practices have shown that an expert can be obtained by RL methods without great effort. In particular, given observation \( \tilde{o} = [o_t, o_v, o_d] \) and the same reward structure \( (N = 1) \) in SecIII-A.2, RL (we use Soft Actor-Critic [27]) can train a navigation expert policy \( \pi_E(\tilde{o}) \) for a single robot efficiently. Though BC is widely used for pre-training to provide initialization for RL. As a result, pre-training from a navigation expert can not provide a good bootstrap for this problem since it performs undesired behaviors that break the global connectivity. Instead, we incorporate BC into RL training. With a navigation expert \( \pi_E \), a supervised mean square error (MSE) loss function is constructed to minimize the difference with \( \pi_\theta \):

\[
J_{BC}(\theta) = \frac{1}{T} \sum_{t=0}^{T} \| \pi_E(\tilde{\theta}^t) - \pi_\theta(\theta^t) \|^2
\]  

(9)

Then the original RL objective Eq. (1) is modified with this additional BC loss term:

\[
\hat{J}(\theta) = J(\theta) - \lambda_c J_{BC}(\theta)
\]  

(10)

where \( \lambda_c > 0 \) is a hyper-parameter.

As a result, the robots attempt to mimic the expert behaviors when optimizing the objective Eq. (10) but eventually surpass the performance of expert \( \pi_E \) due to improvements from the policy gradients of the RL term Eq. (1).

D. Neural network structure

We follow the common practices [9, 10, 17] in the parameterization of policy \( \pi_\theta \) and adopt a similar network structure as shown in Fig. 3. The raw laser data \( o_i \) is processed by two 1D convolution (Conv1D) layers and a fully connected (FC) layer for feature extraction. The output feature vector is concatenated with the rest of observation vector \( [o_e, o_o, o_p] \) and are fed into two hidden FC layers. Then the velocity output \( a_{mean} \) is obtained from the last FC layer. As suggested by a large scale experimental results [28], we use hyperbolic tangent \((\tanh)\) as activation functions and orthogonal initialization scheme to initialize network weights. The output \( a_{mean} \) is supposed to be the mean of the action distribution. A separate vector \( a_{logstd} \) with the same dimension as \( a_{mean} \) is defined as the log standard deviation that will be updated alone. In this way, the stochastic policy \( \pi_\theta \) is formed as a Gaussian distribution \( \mathcal{N}(\text{mean} = a_{mean}, \text{std} = \exp(a_{logstd})) \). At each timestep \( t \), the robots take random actions \( a^t \sim \pi_\theta(\theta^t) \) for on-policy exploration.

The value networks \( V_\phi \) and \( V_\phi \) are introduced for advantage estimation and both constructed with two hidden and one output FC layers to map the observation \( o \) to value estimates.
Algorithm 1 Policy Training with CPO and BC
1: Apply orthogonal initialization to policy network \( \pi_{\theta} \) and value networks \( V_{\phi}, \hat{V}_{\phi} \), provide expert policy \( \pi_{E} \).
2: \( \psi_s = 1 \)
3: for episode \( \psi = 1, 2, \ldots \) do
4: Receive initialized observations \( \{o^{i,t}\}_{i=1}^{N} \) from the environment
5: for robot \( i = 1, 2, \ldots, N \) do
6: Take action \( a^{i,t} \sim \pi_{\theta}(o^{i,t}) \)
7: Receive reward \( r^{i,t} \) and cost signal \( c^{i} \)
8: Repeat 6 and 7, collect transition data \( (o^{i,t}, a^{i,t}, r^{i,t}, c^{i}) \) for \( T_{\psi} \) timesteps
9: end for
10: if \( \sum_{j=\psi, T_j > N_{batch}} \) then
11: // Update \( \pi_{\theta} \) and \( V_{\phi}, \hat{V}_{\phi} \) in batch
12: Compute advantage estimates for batch data
13: Compute BC loss term Eq. (9) with expert policy \( \pi_{E} \) and add to the surrogate objective
14: Update \( \theta \) by approximately solving CPO in Eq. (8)
15: Update \( \phi, \phi_{c} \) by minimizing Eq. (11) with L-BFGS [31]
16: end if
17: \( \psi_s = \psi \)
18: end for

networks \( V_{\phi} \) is optimized with the following loss function:

\[
L_{V}(\phi) = \sum_{i=1}^{N} \left\| V_{\phi}(s_{i}) - \sum_{t=0}^{\infty} \gamma^t r(s_{t+1}, a_{t+1}) \right\|^2
\]

Eq. (11) is computed with all sampled trajectories and minimized with an L-BFGS optimizer. The cost value network \( \hat{V}_{\phi} \) is updated in the same fashion in terms of cost signals.

We implement our neural network models in PyTorch. All training procedures and simulation experiments are performed on a computer with an Nvidia RTX 2080 GPU and an Intel Xeon Gold 6254 CPU. Each timestep takes about 2 ms in various simulations. The primal hyper-parameters for Algorithm 1 are shown in Tab. 1.

TABLE I: Key parameter settings for training

| Parameter | Description               | Value       |
|-----------|---------------------------|-------------|
| \( N_{\text{batch}} \) | number of timesteps for each update | 2048        |
| \( \gamma \) | discount factor for reward | 0.99        |
| \( \gamma_{c} \) | discount factor for cost | 0.999       |
| \( \lambda_{c} \) | coefficient for BC loss | 0.1         |
| \( lr_{\phi}, lr_{\phi_{c}} \) | L-BFGS step size for \( \phi, \hat{\phi}_{c} \) | 0.1         |
| \( \eta \) | KL constraint threshold | 0.01        |
| \( d \) | connectivity constraint threshold | 0.1        |

IV. EXPERIMENTS AND RESULTS

To demonstrate the effectiveness of our approach, we first investigate the influences of behavior cloning and the connectivity constraint on policy performance. Then we evaluate the performance of our approach in real-world experiments.

TABLE II: Evaluation of generalization performance on various approaches with the different number of robots. The results are evaluated in 100 unseen scenarios.

| Metrics          | Approach | 3 robots | 4 robots | 5 robots |
|------------------|----------|----------|----------|----------|
| Success Rate     | CPO      | 0.05     | 0.02     | 0.03     |
|                  | TRPO+BC  | 0.9      | 0.95     | 0.95     |
|                  | CPO+BC   | 0.87     | 0.88     | 0.82     |
| Connectivity Rate| CPO      | 0.07     | 0.3      | 0.07     |
|                  | TRPO+BC  | 0.73     | 0.77     | 0.71     |
|                  | CPO+BC   | 0.73     | 0.77     | 0.71     |
| Travel Time      | CPO      | 16.68    | 14.732   | 10.307   |
|                  | TRPO+BC  | 7.853    | 9.564    | 11.421   |
|                  | CPO+BC   | 7.28     | 13.697   | 9.163    |

A. Comparisons on various approaches in simulation

To study the influences of behavior cloning and connectivity constraint in our approach, we compare three approaches with different combinations:

- **CPO**: policy optimization under the connectivity constraint without supervised BC loss term, i.e., the BC loss coefficient \( \lambda_{c} = 0 \)
- **TRPO+BC**: unconstrained policy optimization with supervised BC loss term where the connectivity constraint is removed.
- **CPO+BC (our approach)**: policy optimization under the connectivity constraint and the supervised BC loss term

The three approaches are compared in terms of average rewards and constraint value during training. To quantitatively measure the final performance of the approaches, we use the following metrics:

- **Success Rate**: the ratio of the robot team to reach targets without collisions
- **Connectivity Rate**: the ratio of the robot team to maintain connectivity during navigation
- **Travel Time**: the average travel time for the robot team successfully navigates to targets

We randomly re-generate 100 maps for final performance evaluation. We conduct experiments with the various number of robots in simulation and evaluate the performance on average with 5 random seeds for each approach. The evolution of learning curves is consistent across different simulation settings, and we present one of the results (3 robots) in Fig. 5a and Fig. 5b.

Influence of behavior cloning: Comparing CPO and CPO+BC shows that behavior cloning help to improve the policy convergence speed, as shown in Fig. 5a. In fact, CPO fails to learn to navigate in most of the training maps because of influences of the connectivity constraint. This is supported by the constraint value evolution over time in Fig. 5b. As shown in Tab. II, the CPO policy leads to terrible failures in terms of navigation success rate and dedicates to preserve connectivity in testing scenarios, which are undesired behaviors for navigation tasks. CPO+BC stabilizes the constraint value to be close to the threshold and succeeds in the majority of unseen scenarios. We conclude that BC has significant effects on policy improvement.

Influence of connectivity constraint: Comparing CPO+BC and TRPO+BC shows that CPO+BC is inferior in terms
of policy convergence speed. This is due to the fact that CPO+BC learn to satisfy the constraint. However, TRPO+BC leads to high constraint values without bounding the constraint on training convergence. The final performance presented in Tab. II also proves that TRPO+BC fails to maintain connectivity in most cases even it has good navigation performance. Surprisingly, CPO+BC outperforms TRPO+BC at travel time. We believe that it is because CPO+BC learns implicit cooperative schemes as presented in many cases.

Compared to TRPO+BC, CPO+BC leads to sub-optimal navigation success rate but we find that most of the failures are caused where the team struggles in complex scenarios which require great exploration abilities. This result indicates that enforcing connectivity constraints degrades the ability of navigation exploration. Compared with CPO, TRPO allows to take “risky” actions without concerns of constraints satisfaction. Both CPO+BC and TRPO+BC show consistent performance across the various number of robots in the experiment results listed in Tab. II.

B. Real-world experiments

As CPO+BC and TRPO+BC have good navigation performance in simulation, we deploy the policies trained by both methods to evaluate their sim-to-real capabilities. The deployed policies take around 2M training timesteps.

*Real robot configuration:* Our holonomic robot model is shown in Fig. 6. The robot is equipped with DJI Mecanum wheel chassis for omnidirectional mobility and an RPLIDAR A2 laser range scanner. Each LiDAR on a robot is deployed at a different height to detect each other. We evenly pick 90 measurements from every 360° laser scan. The robot is controlled by a Jetson TX2 module that publishes the policy output at 10Hz. For both the simulations and the real-world experiments we set the radius of the robot $R_c = 0.18$m and the maximum moving speed $v_{max} = 0.7$m/s. We set $R_c = 1.2$m to emulate limited communication range. To distribute position information to the robots, we use an OptiTrack global tracking system to capture and inform the states of robots and the target position. The robots receive laser data and position information by subscribing to ROS nodes of LiDAR and OptiTrack.

The policies are tested in various scenarios (shown in Fig. 7) and show consistent performance as in simulation on the three metrics in Tab. II. The experiment results are summarized in Fig. 5c, which shows the value of algebraic connectivity of the robots, as the multi-robot system evolves. Our approach (i.e., CPO+BC) takes less travel time and maintains connectivity at all time while TRPO+BC fails in connectivity perseverance (the value goes to 0).

V. CONCLUSION

In this work, we present an RL-based approach to develop a decentralized policy for multi-robot navigation with connectivity maintenance. The policy is shared among robots and jointly optimized by transition data of all robots in simulation during training. By enforcing the connectivity constraint and providing expert demonstrations, the policy learns to generate navigation actions with connectivity concerns. We compare different combinations of our key designs and validate the effectiveness of our approach. We also evaluate the generalization performance of our approach in both various simulated scenarios and real-world experiments. As a case study of RL-based approach, this work presents an effective way to reduce exploration complexity with concerns of constraints satisfaction.
