Connectivity Guaranteed Multi-robot Navigation via Deep Reinforcement Learning

Juntong Lin, Xuyun Yang, Peiwei Zheng, Hui Cheng*
School of Data and Computer Science
Sun Yat-sen University
Guangzhou, Guangdong Province, China
*Corresponding author: chengh9@mail.sysu.edu.cn

Abstract: This paper considers the multi-robot navigation problem where the geometric center of a multi-robot team aims to efficiently reach the waypoint without collisions in unknown complex environments while maintaining connectivity during the navigation. A novel Deep Reinforcement Learning (DRL)-based approach is proposed to derive end-to-end policies for the multi-robot navigation problem. In order to guarantee the connectivity during the navigation, a *constraint satisfying parametric function* (CSPF) is proposed to represent the navigation policy. *Virtual policy extended environment* (VP2E), an implementation framework of the CSPF is accompanied so as to make CSPF compatible with existing DRL techniques which rely on differentiable parametric functions. Both simulations and real-world experiments of a team of 3 holonomic robots are conducted to verify the effectiveness of the proposed DRL-based navigation method.

Keywords: Multi-robot System, Navigation, Deep Reinforcement Learning

1 Introduction

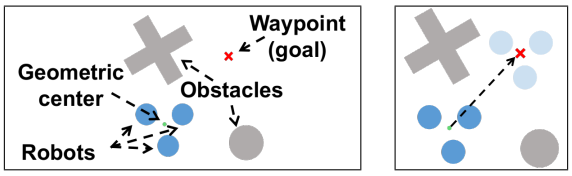
Multi-robot teams have broad applications in surveillance, search, exploration, agricultural spraying, collaborative transportation, etc [1]. When carrying out a mission, a multi-robot team may need to operate in unknown complex environments. Therefore, the navigation policy is essential for the safety and efficiency of the multi-robot team. The communication range is usually limited, thus the navigation policy should take connectivity into consideration so as to guarantee communication for collaborative operations.

Applying Deep Reinforcement Learning (DRL) to find the navigation policy for multi-robot teams seems to be promising regarding the huge success of DRL in various domains including games, robotics, natural language processing, etc [2, 3]. In comparison with conventional rule-based navigation methods [4, 5, 6, 7, 8, 9, 10], DRL-based approaches [11, 12, 13] can derive end-to-end policies which directly map raw sensor data to control signals. Such property is appealing because it removes the necessity of constructing obstacle maps, which is required by conventional rule-based methods. The real-time mapping is challenging and computationally prohibitive sometimes.

In spite of the end-to-end property, most of the existing DRL-based navigation approaches for multi-robot team [11, 12, 13] lose control of the derived policies and can hardly provide constraint guaranteeing policies (in comparison with conventional rule-based methods which can easily provide constraint satisfying policies [4, 5, 6, 7, 8, 9, 10]). The outputs of DRL-based policies (i.e., control signals) are usually unconstrained and therefore always have the chance to break the connectivity constraint, which may lead to undesirable consequences (e.g., communication termination).

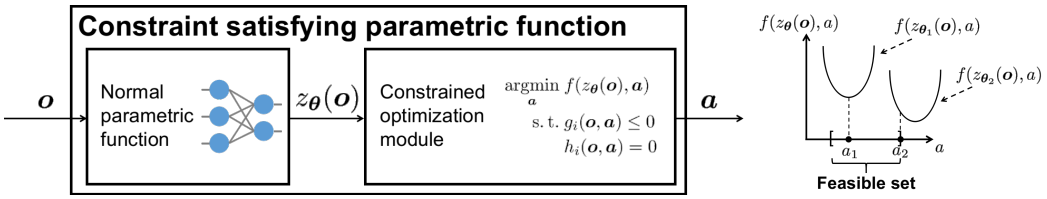
In this paper, we study the multi-robot navigation problem where the geometric center of a multi-robot team aims to efficiently reach the waypoint without collisions in unknown complex environments as shown in Figure 1. During the navigation, the multi-robot team is required to maintain connectivity. We propose a novel DRL-based approach where the navigation policy is represented by the *constraint satisfying parametric function* (CSPF). Consisting of a normal parametric function (e.g., neural network) and a constrained optimization module as shown in Figure 2a, the CSPF allows imposing constraints on the outputs of the navigation policy (i.e., control signals) and therefore can guarantee connectivity during the navigation process. An implementation framework of the

3rd Conference on Robot Learning (CoRL 2019), Osaka, Japan.



(a) (b)

Figure 1: Problem description. (a) Annotations of objects. (b) Navigate the geometric center of the robot team to the waypoint without collisions while maintaining connectivity.



(a) (b)

Figure 2: (a) Constraint satisfying parametric function (CSPF). Consisting of a normal parametric function and a constrained optimization module, CSPF synthesizes representation capacity and constraint guarantee. (b) A toy example (where the output $\boldsymbol{a} \in \mathbb{R}^1$) illustrating the CSPF. Given an input $\boldsymbol{o}$, different parameters $\boldsymbol{\theta}$ correspond to different objective functions and thus produce different final outputs a. Note that the final outputs always lie in the feasible set and satisfy constraints.

CSPF called *virtual policy extended environment* (VP2E) is proposed to make the CSPF compatible with existing DRL techniques which usually rely on differentiable parametric functions [14, 15, 16].

1.1 Related works

In order to strictly guarantee connectivity, a mechanism of imposing constraints on the control signal is required. A prevailing paradigm is reward shaping which either intuitively increases the punishment on undesirable behaviors [11, 12, 13] or derives surrogate reward functions with theoretic foundations [17]. However, reward shaping can only alleviate the violation of constraints but cannot guarantee constraint satisfaction since the control signals are still from unconstrained policies.

In some works, a normalization function (e.g., sigmoid, tanh, clipping) is utilized to force the outputs of parametric function lying in a specific interval [11, 15, 18]. The final control signals are obtained by rescaling the normalized outputs to the target interval. The normalization-based methods are suitable to handle output interval constraints but can hardly cope with constraints like connectivity.

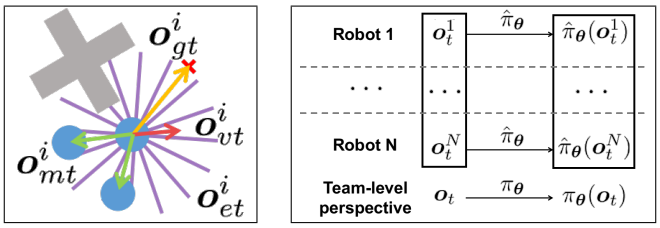
Tools in control theory like control barrier functions [19] and lyapunov functions [20] are also utilized to impose constraints. Such methods have solid theoretic foundations yet introduce additional assumptions (e.g., availability of a valid set [19] and discrete action spaces [20]).

Hierarchical architectures follow the divide and conquer mechanism [21, 22, 23, 24]. In particular, the DRL agent is only responsible for making high-level decisions while the final control signal is left to a low-level constraint satisfying controller. However, designing the hierarchical architecture (i.e., deciding concrete level/meaning of the high-level actions) is nontrivial and challenging sometimes. Moreover, the low-level constraint satisfying controller may be unavailable.

In this paper, CSPF is proposed to represent the navigation policy in order to guarantee the connectivity constraint of multi-robot teams during navigation. In comparison with hierarchical architectures, the CSPF can ensure connectivity while remaining plug-and-play (i.e., no necessity of explicitly designing hierarchical architectures and no requirement on availability of low-level constraint satisfying controllers).

1.2 Contributions and paper organization

The main contributions of this paper are as follows:



(a) (b)

Figure 3: A DRL-based navigation approach [11]. (a) The local observation includes information of the environment (purple lines), teammates (green arrows), waypoint (the yellow arrow) and current velocity (the red arrow). (b) The centralized learning and decentralized execution mechanism.

- A novel DRL-based navigation approach where the navigation policy is represented by CSPF is proposed to navigate the multi-robot team through unknown complex environments. The derived policy can directly map raw sensor data to control velocities without obstacle maps while guaranteeing the connectivity of the multi-robot team;
- VP2E, an implementation framework of CSPF, is proposed to make CSPF compatible with existing DRL techniques which rely on differentiable parametric functions;
- Simulation results illustrate that the multi-robot team can navigate through unknown complex environments without collisions while maintaining connectivity;
- Indoor real-world experiments show that the proposed method succeeds in navigating a team of 3 holonomic unmanned ground vehicles (UGVs) through dense obstacles.

The remainder of this paper is organized as follows: In Section 2, backgrounds including the problem description and a DRL-based navigation method are introduced. In Section 3, our DRL-based multi-robot navigation method with CSPF is proposed. In Section 4, both simulation and real-world experiment results are presented to verify the effectiveness of the proposed method. Section 5 concludes the paper.

2 Backgrounds

In this section, the multi-robot navigation problem is elaborated and a DRL-based multi-robot navigation approach (from which our method is extended) is introduced.

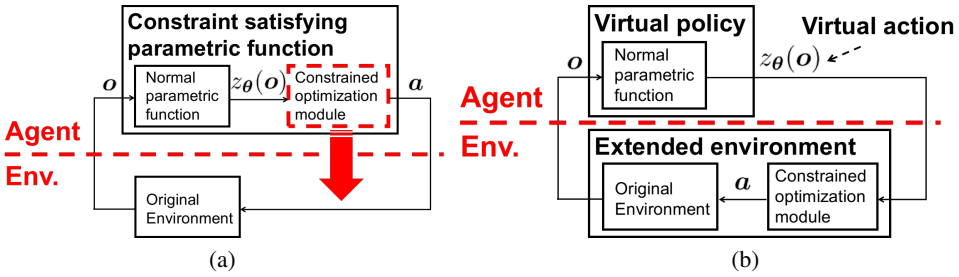
2.1 Problem description

Consider the scenario where a team of N holonomic robots with equal radii R operates in an unkonwn environment containing M static obstacles as shown in Figure 1. The j-th ($1 \leq j \leq M$) obstacle is a polygon denoted as $\tilde{\boldsymbol{p}}^j = [\tilde{\boldsymbol{p}}^j_1, \ldots, \tilde{\boldsymbol{p}}^j_{z_j}]$, where $\tilde{\boldsymbol{p}}^j_{z_j}$ is the position of the z_j-th vertex of the polygon and z_j is the number of vertexes of the j-th obstacle.

At each time step t, the i-th ($1 \leq i \leq N$) robot located at $\boldsymbol{p}^i_t$ with velocity $\boldsymbol{v}^i_t$ receives an observation $\boldsymbol{o}^i_t$ (will be elaborated later) from the environment. Based on the team-level observation $\boldsymbol{o}_t = [\boldsymbol{o}^1_t, \ldots, \boldsymbol{o}^N_t]$, the robot team samples a team-level action (i.e., velocity) $\boldsymbol{a}_t = [\boldsymbol{a}^1_t, \ldots, \boldsymbol{a}^N_t]$ from the distribution $\pi_{\boldsymbol{\theta}}(\boldsymbol{o}_t)$. The distribution $\pi_{\boldsymbol{\theta}}(\boldsymbol{o}_t)$ is generated by the team-level policy $\pi_{\boldsymbol{\theta}}$ which is a mapping from observation to distribution (e.g., Gaussian distribution) parametrized by $\boldsymbol{\theta}$. Within the time interval Δt, position of each robot is updated according to the corresponding action $\boldsymbol{a}^i_t$.

As shown in Figure 3a, the observation $\boldsymbol{o}^i_t$ for the i-th robot at time step t includes information about environment $\boldsymbol{o}^i_{et}$ (i.e., lidar), teammates $\boldsymbol{o}^i_{mt} = [\boldsymbol{p}^1_t - \boldsymbol{p}^i_t, \ldots, \boldsymbol{p}^{i-1}_t - \boldsymbol{p}^i_t, \boldsymbol{p}^{i+1}_t - \boldsymbol{p}^i_t, \ldots, \boldsymbol{p}^N_t - \boldsymbol{p}^i_t]$, goal $\boldsymbol{o}^i_{gt} = \boldsymbol{g} - \boldsymbol{p}^i_t$ and current velocity $\boldsymbol{o}^i_{vt}$.

The multi-robot navigation problem aims to derive a navigation policy $\pi_{\boldsymbol{\theta}}$ which efficiently guides the geometric center of the robot team $\bar{\boldsymbol{p}}_t = \sum_{i=1}^N \boldsymbol{p}^i_t$ to a waypoint/goal $\boldsymbol{g}$ provided by global planners without collisions. During the navigation, the robot team should maintain connectivity (i.e., $\| \boldsymbol{p}^i_t - \boldsymbol{p}^j_t \|_2 \leq d$ for any robot $i \neq j$, where d is the communication range).



(a) (b)

Figure 4: Virtual policy extended environment (VP2E). (a) Original DRL problem with undifferentiable CSPF and original environment. (b) An equivalent DRL problem with differentiable virtual policy and extended environment.

2.2 A DRL-based approach for multi-robot navigation

A DRL-based approach is proposed in the work [11] to solve the multi-robot navigation problem presented above. The DRL-based approach formulates the multi-robot navigation problem as a partial observable Markov decision process (POMDP) and derives a decentralized navigation policy by a centralized learning decentralized execution mechanism.

The work proposes to adopt a centralized learning and decentralized execution mechanism where decentralized navigation policy for multi-robot navigation problem can be derived by DRL techniques originally intended for single-agent (robot) problem. As shown in Figure 3b, the main idea of the mechanism in the work [11] is imposing constraints on the team-level policy $\pi_{\boldsymbol{\theta}}$. In particular, the team-level policy is restricted to be a concatenation of robot-level policies and the parameters of robot-level policies are shared (i.e., $\pi_{\boldsymbol{\theta}}(\boldsymbol{o}_t) = [\hat{\pi}_{\boldsymbol{\theta}}(\boldsymbol{o}_t^1), \hat{\pi}_{\boldsymbol{\theta}}(\boldsymbol{o}_t^2), \ldots, \hat{\pi}_{\boldsymbol{\theta}}(\boldsymbol{o}_t^N)]$, where $\hat{\pi}_{\boldsymbol{\theta}}$ is the robot-level policy). Viewing the robot team as an entity, the team-level policy $\pi_{\boldsymbol{\theta}}$ can be derived by DRL algorithms originally intended for single agent (e.g., Proximal Policy Optimization [14]). In the meanwhile, the derived team-level policy $\pi_{\boldsymbol{\theta}}$ can be executed in a decentralized manner since the underlying robot-level policy $\hat{\pi}_{\boldsymbol{\theta}}$ only takes robot-level observation $\boldsymbol{o}_t^i$ as input. One may refer to the work [11] for more details.

3 Methodology

In this section, we first present the CSPF accompanied with VP2E (an implementation framework making CSPF compatible with existing DRL techniques which rely on differentiable parametric functions) and then elaborate the application of CSPF in multi-robot navigation problem.

3.1 Constraint satisfying parametric function

Commonly used parametric functions in DRL (e.g., neural networks) have powerful representation capacity. However, those parametric functions are unconstrained and may therefore produce outputs (i.e., control signals) which induce violation of constraints. In the context of multi-robot navigation, such violation of constraints may be the break of connectivity among robots. Aiming to remedy the limitation (i.e., possible constraint violations) of common parametric functions, we propose to represent the navigation policy by CSPF.

As shown in Figure 2a, CSPF consists of a normal parametric function and a constrained optimization module. In comparison with common parametric function, the outputs of CSPF are not directly from the normal parametric function $z_{\boldsymbol{\theta}}$ but from the constrained optimization module. Therefore, the outputs are no longer unconstrained and may not induce violation of constraints. In CSPF, the outputs of normal parametric function $z_{\boldsymbol{\theta}}(\boldsymbol{o})$ are interpreted as the inputs of the objective functions $f(\boldsymbol{z}, \boldsymbol{a})$ in the constrained optimization module. The final control signals $\boldsymbol{a}$ are variables to be optimized in the constrained optimization module, and thus can be imposed constraints. To illustrate the core idea of CSPF, a toy example is presented in Figure 2b. Given an input $\boldsymbol{o}$, different parameters $\boldsymbol{\theta}$ correspond to different objective functions and thus produce different final outputs a. Note that the final outputs always lie in the feasible set and satisfy constraints.

Mathematically, the CSPF described above is a mapping from input $\boldsymbol{o} \in \mathbb{R}^o$ to output $\boldsymbol{a} \in \mathbb{R}^a$ which is parametrized by $\boldsymbol{\theta}$ and can be expressed as a contrained optimization problem as follows:

$$\begin{aligned} \underset{\boldsymbol{a}}{\operatorname{argmin}}\ & f(z_{\boldsymbol{\theta}}(\boldsymbol{o}), \boldsymbol{a}) \\ \text{s. t. } & g_i(\boldsymbol{o}, \boldsymbol{a}) \leq 0, \quad \text{for } i = 1, 2, \ldots, m \\ & h_i(\boldsymbol{o}, \boldsymbol{a}) = 0, \quad \text{for } i = 1, 2, \ldots, n, \end{aligned} \tag{1}$$

where $z_{\boldsymbol{\theta}}(\boldsymbol{o})$ is a normal parametric function (e.g., neural network) parametrized by $\boldsymbol{\theta}$, $f(\boldsymbol{z}, \boldsymbol{a})$ is a predefined objective function (e.g., quadratic function) taking the output of the normal parametric function $\boldsymbol{z}$ and the final output $\boldsymbol{a}$ as input, $g_i(\boldsymbol{o}, \boldsymbol{a})$ and $h_i(\boldsymbol{o}, \boldsymbol{a})$ are functions for imposing constraints on the final output $\boldsymbol{a}$.

With the predefined objective function $f(\boldsymbol{z}, \boldsymbol{a})$ serving as a bridge, CSPF takes the advantages of representation capacity from normal parametric functions and constraint guarantee property from the constrained optimization module. Though the concrete form of objective function $f(\boldsymbol{z}, \boldsymbol{a})$ needs to be predefined in advance manually, we think this process is acceptable and can be analogized to the common process of designing architectures of neural networks.

It should be pointed out that instead of first training a neural network during training and then appending a constraint layer to the neural network during execution, we consistently regard the whole CSPF as the policy during both training and execution.

Note that even given a differentiable parametric function $z_{\boldsymbol{\theta}}$, the CSPF in Equation 1 may become undifferentiable because of the constrained optimization module and thus incompatible with those DRL techniques relying on differentiable parametric functions [14, 15, 16]. In order to cope with the possible incompatibility induced by undifferentiability and leverage the powerful existing DRL techniques, we propose VP2E, an implementation framework of CSPF.

In VP2E, the separation between agent and environment no longer lies between CSPF and the original environment as shown in Figure 4a but rather between the *virtual policy* and the *extended environment* as shown in Figure 4b. In particular, the *virtual policy* refers to the normal differentiable parametric function (e.g., neural network) producing virtual actions (i.e., inputs of the objective function in Equation 1) for the extended environment. The *extended environment* stacks the constrained optimization problem on the original environment (which can be easily implemented by a preprocessing module) and thus becomes an environment takes virtual actions as inputs. By implementing the CSPF in the VP2E framework, the original DRL problem (with possible undifferentiable CSPFs and original environments) is transformed into an equivalent DRL problem (with **differentiable** virtual policies and extended environments) and thus can leverage existing powerful DRL techniques which rely on differentiable parametric functions.

3.2 Connectivity guaranteed multi-robot navigation

In this part, we demonstrate how to extend the DRL-based approach in the work [11] with the proposed CSPF to derive connectivity guaranteeing navigation policies. In particular, we retain the overall framework of the DRL-based method in the work [11] and substitute the CSPF for the original unconstrained parametric function (a neural network). More specifically, the neural network originally representing the navigation policy $\pi_{\boldsymbol{\theta}}$ now serves as the $z_{\boldsymbol{\theta}}$ in Equation 1 and its network architecture remains unchanged. In this paper, the concrete form of objective function in Equation 1 is designed to be $f(\boldsymbol{z}, \boldsymbol{a}) = \boldsymbol{a}^T\boldsymbol{a} + \boldsymbol{z}^T\boldsymbol{a}$ for simplicity. Note that the choice of objective function is not unique and is up to the specific applications.

We assume the kinematic model of the robots are known and its mathematical expression is $\boldsymbol{p}_{t+1}^i = \boldsymbol{p}_t^i + \boldsymbol{a}_t^i\,\Delta t$, where $\boldsymbol{p}_t^i/\boldsymbol{a}_t^i$ are the position/action of the i-th robot at time step t. Note that the assumption of known kinematic model is completely different from the assumption of known dynamics $P(\boldsymbol{o}_{t+1}^i|\boldsymbol{o}_t^i, \boldsymbol{a}_t^i)$. The former is the description of the position transition, while the latter is the description of the observation transition (which is hard to obtain since the observation $\boldsymbol{o}_t^i$ contains external environment information such as sensing data).

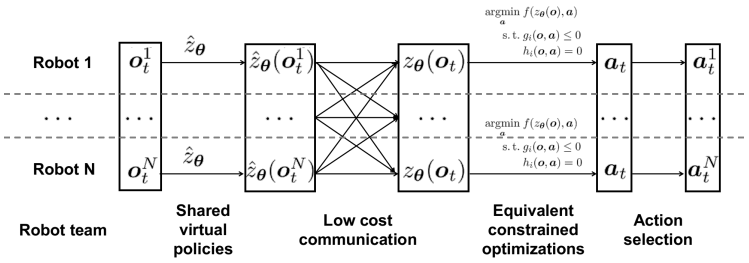


Figure 5: Connectivity guaranteed multi-robot navigation. The i-th robot computes its final control signal $\boldsymbol{a}_t^i$ based on its local observation $\boldsymbol{o}_t^i$ by sequentially running the virtual policies $\hat{z}_{\boldsymbol{\theta}}$, sharing intermediate results $\hat{z}_{\boldsymbol{\theta}}(\boldsymbol{o}_t^i)$ (low cost communications of 2D vectors), solving a constrained optimization problem to obtain $\boldsymbol{a}_t$ and selecting the corresponding action $\boldsymbol{a}_t^i$.

Given the kinematic model, the connectivity constraint can be imposed to the control signal $\boldsymbol{a}_t$ with constraint functions in Equation 1 and the corresponding CSPF is formulated as follows:

$$\begin{aligned} &\underset{\boldsymbol{a}_t}{\operatorname{argmin}}\ \boldsymbol{a}_t^T \boldsymbol{a}_t + z_{\boldsymbol{\theta}}(\boldsymbol{o}_t)^T \boldsymbol{a}_t \\ &\quad \text{s.t. } \|\boldsymbol{p}_{t+1}^i - \boldsymbol{p}_{t+1}^j\|_2^2 - d^2 \le 0 \quad \text{for } 1 \le i < j \le N \quad \text{// connectivity contraints} \\ &\qquad \boldsymbol{p}_{t+1}^i \ = \ \boldsymbol{p}_t^i + \boldsymbol{a}_t^i \, \Delta t, \quad \text{for } i = 1, 2, \ldots, N. \quad \text{// kinematic model} \end{aligned} \tag{2}$$

We assume the connectivity constraint is satisfied at the initial time step (i.e., $\|\boldsymbol{p}_0^i - \boldsymbol{p}_0^j\|_2^2 - d^2 \le 0$). As a result, the feasible set of Equation 2 (i.e., the set of actions guaranteeing the connectivity constraint at the next time step) is always nonempty, which can be easily proved by the mathematical induction: suppose the connectivity constraint is satisfied at time step t (i.e., $\|\boldsymbol{p}_t^i - \boldsymbol{p}_t^j\|_2^2 - d^2 \le 0$), there always exists a solution $\boldsymbol{a}_t = \boldsymbol{0}$ to guarantee the connectivity constraint at time step $t+1$. The connectivity constraint at the initial time step is satisfied. Therefore, there always exists a solution to guarantee the connectivity constraint, i.e., the feasible set is nonempty.

The Equation 2 can be further transformed by substituting the kinematic model and utilizing the following equation (which is the definition of the teammate information in observation $\boldsymbol{o}_{mt}^i$):

$$\boldsymbol{o}_{mt}^{ij} = \begin{cases} \boldsymbol{p}_t^j - \boldsymbol{p}_t^i & \text{for } j < i \\ \boldsymbol{p}_t^{j+1} - \boldsymbol{p}_t^i & \text{otherwise.} \end{cases} \tag{3}$$

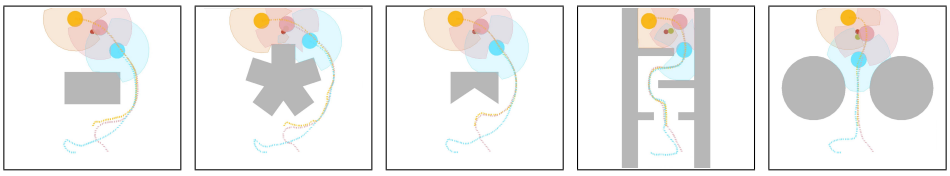
We take the 1-st robot as an example to show the process of decision making. The final mathematical expression of CSPF from the perspective of the 1-st robot (i.e., transform the Equation 2 with the teammate observation $\boldsymbol{o}_{mt}^1$ of the 1-th robot) is as follows:

$$\begin{aligned} &\underset{\boldsymbol{a}_t}{\operatorname{argmin}}\ \boldsymbol{a}_t^T \boldsymbol{a}_t + z_{\boldsymbol{\theta}}(\boldsymbol{o}_t)^T \boldsymbol{a}_t \\ &\quad \text{s.t. } (\Delta t)^2 \|\boldsymbol{a}_t^{i+1} - \boldsymbol{a}_t^1\|_2^2 + 2\Delta t (\boldsymbol{o}_{mt}^{1i})^T (\boldsymbol{a}_t^{i+1} - \boldsymbol{a}_t^1) + \|\boldsymbol{o}_{mt}^{1i}\|_2^2 \le d^2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{for } i = 1, \ldots, N-1 \\ &\qquad (\Delta t)^2 \|\boldsymbol{a}_t^{i+1} - \boldsymbol{a}_t^{j+1}\|_2^2 + 2\Delta t (\boldsymbol{o}_{mt}^{1i} - \boldsymbol{o}_{mt}^{1j})^T (\boldsymbol{a}_t^{i+1} - \boldsymbol{a}_t^{j+1}) + \|\boldsymbol{o}_{mt}^{1i} - \boldsymbol{o}_{mt}^{1j}\|_2^2 \le d^2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{for } 1 \le i < j \le N-1. \end{aligned} \tag{4}$$

The Equation 4 is a convex optimization problem on variable $\boldsymbol{a}_t$ and thus can be solved efficiently. In this work, we use SNOPT [25] to solve the Equation 4. After solving the optimization problem in Equation 4, the 1-st robot should select its corresponding action $\boldsymbol{a}_t^1$ in the result $\boldsymbol{a}_t$ to execute. To summarize, the overall process of the proposed connectivity guaranteed multi-robot navigation is shown in Figure 5. Note that the transformation from Equation 2 to Equation 4 is not unique. Each robot can derive an equivalent constrained optimization problem by utilizing its own teammate observation $\boldsymbol{o}_{mt}^i$.

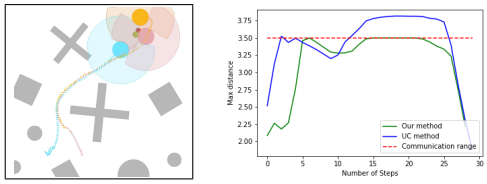
4 Experiments

In this section, simulation and real-world experimental results with a team of 3 holonomic robots are presented to verify the effectiveness of the proposed DRL-based navigation method.



(a) Block. (b) Gear. (c) Groove. (d) Corridor. (e) Passageway.

Figure 6: Qualitative simulation results. Obstacles are grey polygons. Robots are pink, brown and blue circles. Shaded discs are lidar measurements. Green circle is the centroid of the robot team. The goal is the dark red circle and colored dashed lines are the trajectories of robots with corresponding colors. The color scheme applies across all simulation results.



(a) Random. (b) Connectivity curve.

Figure 7: An analysis on a navigation process. (a) A randomly generated scenario. (b) The connectivity curve of the navigation in (a), where the x-axis is the time step and the y-axis is the maximum distance among robots at a specific time step. The red dashed line is the communication range d.

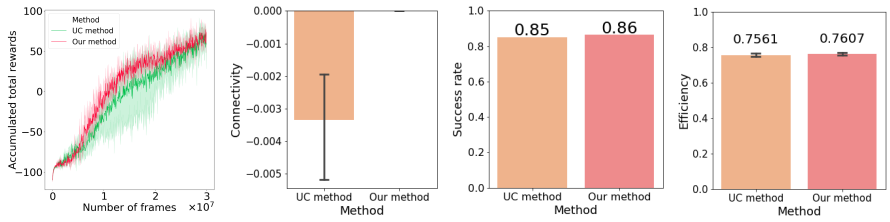
4.1 Simulation results

In this part, we show the performance of the proposed method in terms of the learning process and the execution period. For simplicity, we refer the work [11] by "UC method" (UC is the abbreviation of unconstrained) in the rest of this paper. We retain the overall training configuration of the UC method and simplify the two period learning with the linear learning rate decay.

Figure 6 shows that our method manages to reach goals in various scenarios including the passageway (Figure 6e) where a rule-based method (more precisely, an artificial potential field method [5]) fails [11]. Moreover, it can be seen from Figure 7 that the connectivity is consistently maintained when navigating with our method while the connectivity constraint is violated during the navigation with the UC method. It shows the advantage of our method in terms of guaranteeing connectivity.

The training curve is shown in Figure 8a. It can be seen that the convergence rate is enhanced and the variance is reduced after applying CSPF to guarantee connectivity. The improvement is reasonable since the connectivity constraint can be viewed as a prior knowledge incorporated into the learning process. In comparison with policies represented by unconstrained parametric functions in the UC method, policies in our method are provided with the prior knowledge "don't break the connectivity" via the CSPF and thus constantly guarantee connectivity, which frees the DRL agent from wasting time on learning the behavior of maintaining connectivity. As a consequence, the agent can concentrate on those behaviors which have to be learned in a trial-and-error manner and achieves a better performance.

To evaluate the performance of the proposed method, we conduct a statistical comparison where 100 scenarios are generated randomly (look like Figure 7a) to test the derived policies. The adopted quantitative metrics are defined as follows [11]:

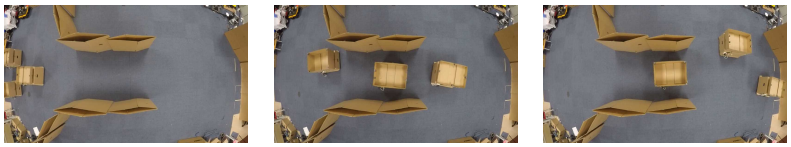


(a) Convergence curve. (b) Connectivity. (c) Success rate. (d) Efficiency.

Figure 8: Quantitative simulation results.

(a) Chasis. (b) Appearance. (c) Robot team.

Figure 9: Real-world experimental platform



(a) Initial positions. (b) Switch formation. (c) Reach the goal.

Figure 10: Snapshots of a team of 3 holonomic UGVs navigating through obstacles.

- Connectivity: The negative average distance constraint violation over a trajectory, i.e., $-(1/T')\sum_{t=1}^{T'} c_t$, where $c_t = \text{avg}(\{\max(0, \|\boldsymbol{p}_t^i - \boldsymbol{p}_t^j\|_2 - d) \mid \forall i \neq j\})$ is the average distance constraint violation at time step t and T' is the trajectory length.
- Success rate: The ratio of the success times $n_{success}$ over the total number of test cases n_{total}, i.e., $n_{success}/n_{total}$.
- Efficiency: The ratio of the travel time lower bound t_{lb} (dividing Euclidean distance between the initial position and the goal by the maximum velocity) over the actual travel time t_{travel}, i.e., t_{lb}/t_{travel}.

Higher values indicate better performance for all quantitative metrics. Failure navigations are neglected in connectivity and efficiency. To alleviate stochastic errors, the final quantitative metrics of a method is the average performance of 3 policies trained with different random seeds. It can be seen that the proposed method can strictly guarantee connectivity (Figure 8b) while achieving comparable performance in success rate (Figure 8c) and efficiency (Figure 8d) compared with the UC method.

4.2 Real-world experimental results

To validate the practicability of the proposed method in real world, we deploy the derived policy of the proposed method on a team of 3 holonomic UGVs. As shown in Figure 9, each UGV is equipped with a 2D lidar (RPLIDAR-A2) and an onboard computer (Nvidia Jetson TX2). Besides, UGVs can communicate virtual polices $\hat{z}_{\boldsymbol{\theta}}(\boldsymbol{o}_t^i)$ between each other. In order to enable detections among UGVs, we wrap UGVs with boxes and set 2D lidars at different heights. Positions and velocities are provided by the OptiTrack motion capture system.

The snapshots of a real-world indoor experiment are presented in Figure 10. With the end-to-end policy derived by the proposed method, the robot team succeeds in navigating through cluttered obstacles without collisions while maintaining connectivity. It shows the practicability of our proposed method. It should be noticed that the kinematic model $\boldsymbol{p}_{t+1}^i = \boldsymbol{p}_t^i + \boldsymbol{a}_t^i \, \Delta t$ (which we assume the robots obey) is relatively imprecise in the real world. The violation of kinematic model assumption will hamper the navigation performance. We attribute the success of real-world deployment to the merits of the closed-loop control (in comparison with open-loop control).

5 Conclusion

This paper presents a DRL-based approach for the multi-robot navigation problem. By extending a previous DRL-based method with the proposed CSPF (accompanied with VP2E), connectivity guaranteed end-to-end navigation policies can be derived. Simulation results show that the CSPF can indeed accelerate the learning process and guarantee connectivity during navigation. The real-world experiments of a team of 3 holonomic UGVs further verify the practicability of the proposed method. In the future work, the proposed method may be further decentralized so as to remove the necessity of explicit data communications.

Acknowledgments

This work is supported by Major Program of Science and Technology Planning Project of Guangdong Province (2017B010116003), NSFC-Shenzhen Robotics Projects (U1613211), and Guangdong Natural Science Foundation (1614050001452, 2017A030310050).

References

[1] H. G. Tanner and A. Kumar. Towards decentralization of multi-robot navigation functions. In *IEEE Int. Conf. on Robotics and Automation*, pages 4132–4137, 2005.

[2] Y. Li. Deep reinforcement learning: An overview. *arXiv:1701.07274*, 2017.

[3] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The Int. Journal of Robotics Research*, 32(11):1238–1274, 2013.

[4] T. J. Koo and S. M. Shahruz. Formation of a group of unmanned aerial vehicles (uavs). In *American Control Conf.*, pages 69–74, 2001.

[5] T. Balch and M. Hybinette. Social potentials for scalable multi-robot formations. In *IEEE Int. Conf. on Robotics and Automation*, pages 73–80, 2000.

[6] A. Wasik, J. N. Pereira, R. Ventura, P. U. Lima, and A. Martinoli. Graph-based distributed control for adaptive multi-robot patrolling through local formation transformation. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1721–1728, 2016.

[7] R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Trans. on Automatic Control*, 51(3):401–420, March 2006.

[8] M. Saska, V. Vonsek, T. Krajnk, and L. Peuil. Coordination and navigation of heterogeneous mavugv formations localized by a hawk-eye-like approach under a model predictive control scheme. *The Int. Journal of Robotics Research*, 33(10):1393–1412, 2014.

[9] J. Alonso-Mora, S. Baker, and D. Rus. Multi-robot navigation in formation via sequential convex programming. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 4634–4641, 2015.

[10] J. Alonso-Mora, E. Montijano, M. Schwager, and D. Rus. Distributed multi-robot formation control among obstacles: A geometric and optimization approach with consensus. In *IEEE Int. Conf. on Robotics and Automation*, pages 5356–5363, 2016.

[11] J. Lin, X. Yang, P. Zheng, and H. Cheng. End-to-end decentralized multi-robot navigation in unknown complex environments via deep reinforcement learning. *arXiv:1907.01713*, 2019.

[12] Z. Xu, Y. Lyu, Q. Pan, J. Hu, C. Zhao, and S. Liu. Multi-vehicle flocking control with deep deterministic policy gradient method. In *IEEE Int. Conf. on Control and Automation*, pages 306–311, 2018.

[13] C. Wang, J. Wang, and X. Zhang. A deep reinforcement learning approach to flocking and navigation of uavs in large-scale complex environments. In *IEEE Global Conf. on Signal and Information Processing*, pages 1228–1232, 2018.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[15] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv:1509.02971*, 2015.

[16] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *Int. Conf. on Machine Learning*, pages 1889–1897, 2015.

[17] J. Achiam, D. Held, A. Tamar, and P. Abbeel. Constrained policy optimization. In *Int. Conf. on Machine Learning*, pages 22–31, 2017.

[18] P. Long, T. Fanl, X. Liao, W. Liu, H. Zhang, and J. Pan. Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning. In *IEEE Int. Conf. on Robotics and Automation*, pages 6252–6259, 2018.

[19] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. *arXiv:1903.08792*, 2019.

[20] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 8092–8101, 2018.

[21] K. Min, H. Kim, and K. Huh. Deep distributional reinforcement learning based high level driving policy determination. *IEEE Trans. on Intelligent Vehicles*, 2019.

[22] H. Banzhaf, P. Sanzenbacher, U. Baumann, and J. M. Zöllner. Learning to predict ego-vehicle poses for sampling-based nonholonomic motion planning. *IEEE Robotics and Automation Letters*, 4(2):1053–1060, 2019.

[23] C. Hubschneider, A. Bauer, J. Doll, M. Weber, S. Klemm, F. Kuhnt, and J. M. Zöllner. Integrating end-to-end learned steering into probabilistic autonomous driving. In *IEEE Int. Conf. on Intelligent Transportation Systems*, pages 1–7, 2017.

[24] S. Shalev-Shwartz, S. Shammah, and A. Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv:1610.03295*, 2016.

[25] P. E. Gill, W. Murray, and M. A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM review*, 47(1):99–131, 2005.