Top-two ListMLE Reinforcement Learning Based UGVs Formation Control with Changeable Pattern

Chengzhi Lei, Qiang Xiao, Zhigang Zeng

***Abstract*— In this paper, we propose a method to achieve reinforcement learning based with a changeable pattern using a top-two selection mechanism. The selection mechanism functions as a method that fixes the number of utilized neighbors by training the scoring network through ListMLE loss computation between the top-two candidates derived from limited neighbor information scoring and the top-two estimates based on global value network. With fixed neighbor utilization, formation errors and other neighbor-related information can be directly fed into the policy network without structural modifications. Unlike other approaches that embed formation constraints in value networks, our method directly inputs formation error into the policy network, enabling formation control with changeable pattern. In the experiments, we further propose a training data processing method to handle varying numbers of neighbors across batches, where neighbor sequences are vectorized by masking techniques improving training efficiency.**

I. Introduction

Formation control, a cornerstone of multi-agent systems, enables autonomous agents like UAVs and UGVs to achieve and maintain desired geometric configurations, with applications spanning industrial, rescue, and military domains. Early methodologies, such as the virtual structures approach by Lewis et al. [1] and the behavior-based framework by Balch and Arkin [2], laid the groundwork for high-precision and adaptive formation control. Recent advancements, including data-driven techniques for systems with unknown dynamics [3], have further expanded the field. Notably, reinforcement learning (RL) has emerged as a powerful tool for formation control due to its model-free nature, enabling agents to learn complex policies without explicit dynamical models. With multi-agent reinforcement learning (MARL) algorithm, RL-based policy can deal with formation control, and some examples: DQN [4], DDPG [5], PPO [6], and momentum policy gradient [7], have demonstrated its success in discrete and continuous action spaces for formation tasks.

A critical challenge in RL-based formation control lies in handling variable numbers of agents. Traditional actor-critic architectures face input dimensionality mismatches when agent counts change, as the critic network must process dynamic system-wide information, and the actor's observation space depends on a fixed number of neighbors. Prior solutions include graph neural networks (GNNs) [8], which parameterize policies independent of graph size, and teacher-student frameworks [6] that enumerate scenarios for different agent counts. However, GNN often requires deep architectures for effective training, while teacher-student methods lack scalability. Furthermore, many RL approaches simplify dynamics of control objects during training, limiting real-world applicability.

To address these limitations, we propose a novel RL-based formation control method with changeable patterns for UGVs, leveraging a top-two selection mechanism to dynamically adapt to varying UGV numbers in actor. Our approach trains a scoring network to rank neighbors using local observations and global value estimates, resolving input dimensionality conflicts by selecting a fixed number of neighbors. The scoring network is optimized via a modified ListMLE [9] loss, which aligns the top-two candidates from local scoring with those from a global value network. This mechanism decouples policy inputs from the total number of UGVs, enabling the policy network to process formation errors and neighbor states without structural modifications. Unlike existing method [6] that encode formation pattern in value networks, our framework directly injects formation errors into the policy network, enhancing flexibility for dynamic pattern changes. For the critic, we use graph convolutional network (GCN) to replace GNN, where fewer layers are required. To further improve training efficiency with variable neighbors, we introduce a masking-based batch processing technique that vectorizes irregular neighbor sequences.

The contributions are listed below

- We propose a top-two selection algorithm trained by ListMLE that selects the two most valuable neighbors and leverage their information to generate action.
- The formation pattern can vary between different tasks, benefiting from the proposed algorithm that fixes the number of neighbors utilized.
- We proposed a way that enable batch processing of neighbors' information across UGVs or parallel environments with varying numbers of neighbors.

II. Problem Formulation

A. Notations

The system in this paper consists of n follower UGVs and one leader UGV, denoted as $\mathcal{V} = \{v_i\},\ i \in \{0, 1, \cdots, n\}$, where $\{1, \cdots, n\}$ is the follower index set, and 0 is the leader index.

The properties of each UGV used for policy are position, velocity, orientation and LiDAR sensor data. The position in world coordinate of ith UGV is denoted as $p_i = [x_i, y_i]^\top$ and acceleration denote as $\mathbf{a}_i = [a_{xi}, a_{yi}]^\top$, where $i \in \{0, 1, \cdots, n\}$. The relative position of jth UGV in the coordinate of ith UGV is denoted as $p_{ij} = p_j - p_i = [x_{ij}, y_{ij}]^\top$. The relative distance between ith UGV and jth UGV is denoted as d_{ij}. The velocity of ith UGV in world

coordinate is denoted as $\mathbf{v}_i = [v_i, \omega_i]^\top$, where v_i is the linear speed and ω_i the angular speed, and the relative velocity of jth UGV in ith UGV's coordinate as $\mathbf{v}_{ij} = [v_{ij}, \omega_{ij}]^\top$. The orientation is defined as the heading of ith UGV, denoted as φ_i. LiDAR is equipped for collision detecting, and the data from LiDAR of ith UGV is denoted as $\mathbf{s}_i = \left[s_1^i, \ldots, s_m^i\right]^\top$, in which $m = r \times 360$ and r represents the resolution per degree. We use dim to indicate the dimension of a vector.

B. Control Objective

We consider a UGV formation system in planar environments employing a leader-follower framework. The configuration comprises a leader UGV capable of executing planned trajectories or maintaining stationary, accompanied by follower UGVs that utilize trained reinforcement learning policy to preserve formation geometry while ensuring inter UGV collision avoidance.

The UGVs employ a differential-drive mechanism with independently actuated wheel pairs, where their global translational and rotational velocities emerge from coordinated control of bilateral wheel speeds as

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{l_x} & -\frac{1}{l_x} \end{bmatrix} \begin{bmatrix} v_r \\ v_l \end{bmatrix}, \tag{1}$$

where l_x is the distance between two wheels, v_l is the linear speed of the left wheel and v_r the right side one. Convert (1) into Cartesian coordinate

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & 0 \\ \sin\theta & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ 0 \\ \omega \end{bmatrix}, \tag{2}$$

where θ is orientation of UGV in global coordinate. For enhanced clarity, Fig. 1 explicitly details the locomotion mechanism of the UGV.

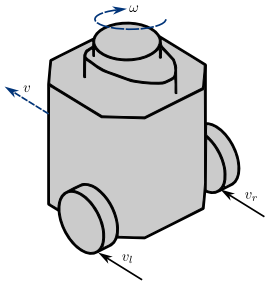


Fig. 1: A UGV locomotion mechanism

The control objective is to minimize the global formation error e_f defined by the distance-based formation, which consists of the sum of individual UGV formation errors e_{if} as:

$$e_{if} = \frac{1}{2} \sum_{j \in \mathcal{N}\{i\}} a_{ij} \left| \|p_{ij}\| - d_{ij}^* \right| + b_i \left| \|p_{i0}\| - d_{i0}^* \right|, \tag{3}$$

$$e_f = \sum_{i=1}^{n} e_{if}. \tag{4}$$

III. Method

A. PPO with Top-two ListMLE

ListMLE is a method for calculating the loss between a target order of items and the estimated order predicted by a model based on scoring functions. The key idea is to measure how well the model's predicted order aligns with the true order by leveraging the Plackett-Luce model, a probabilistic framework for ranking.

The probability of a specific order π under the scores s is defined by the Plackett-Luce model as

$$P(\pi|s) = \prod_{j=1}^{n} \frac{\exp(s_{\pi(j)})}{\sum_{k=j}^{n} \exp(s_{\pi(k)})}, \tag{5}$$

where $\pi(j)$ represents the index of the item in the j-th position of the order, and $s_{\pi(j)}$ is the corresponding score assigned by the model. The model computes the probability of the order π by sequentially selecting items based on their scores, normalized by the scores of the remaining items at each step.

To optimize the model, it aims to minimize the negative log-likelihood of the true order π, which serves as the loss function

$$\mathcal{L} = -\log P(\pi|s) = -\sum_{j=1}^{n} \left[s_{\pi(j)} - \log \left(\sum_{k=j}^{n} \exp(s_{\pi(k)}) \right) \right]. \tag{6}$$

This loss function penalizes deviations between the predicted scores and the true order, guiding the model to produce scores that better align with the desired ranking.

In our formation control scenario, we need to select a fixed number of neighbors and feed the information about the selected neighbors into the policy network, so that the policy network can adapt to changes in the number of neighbors due to adjustments in formation geometry or inherent differences in the number of neighbors among UGVs. Selecting two as the fixed number, so we care about only the top-two valuable neighbors during control. Thus, we edit the probability that

$$P(\pi|s)_{\text{top-two}} = \prod_{j=1}^{2} \frac{\exp(s_{\pi(j)})}{\sum_{k=j}^{n} \exp(s_{\pi(k)})}. \tag{7}$$

Comparing to the original Plackett-Luce model (5), in this modified version, only possibilities of the order of the top-two in π are considered, which reduces the computational complexity of the loss calculation while meet the requirement of selecting a fixed number of two neighbors in our formation control scenario. Based on (7) the loss function of top-two ListMLE is

$$\mathcal{L}_{=} -\log P(\pi|s) = -\sum_{j=1}^{2} \left[s_{\pi(j)} - \log \left(\sum_{k=j}^{n} \exp(s_{\pi(k)}) \right) \right]. \tag{8}$$

PPO is used as the basic algorithm, the model structure is specified in Fig. 2. The scoring function is an MLP layer as the MLP1 shown in Fig. 2, and the target order is defined by

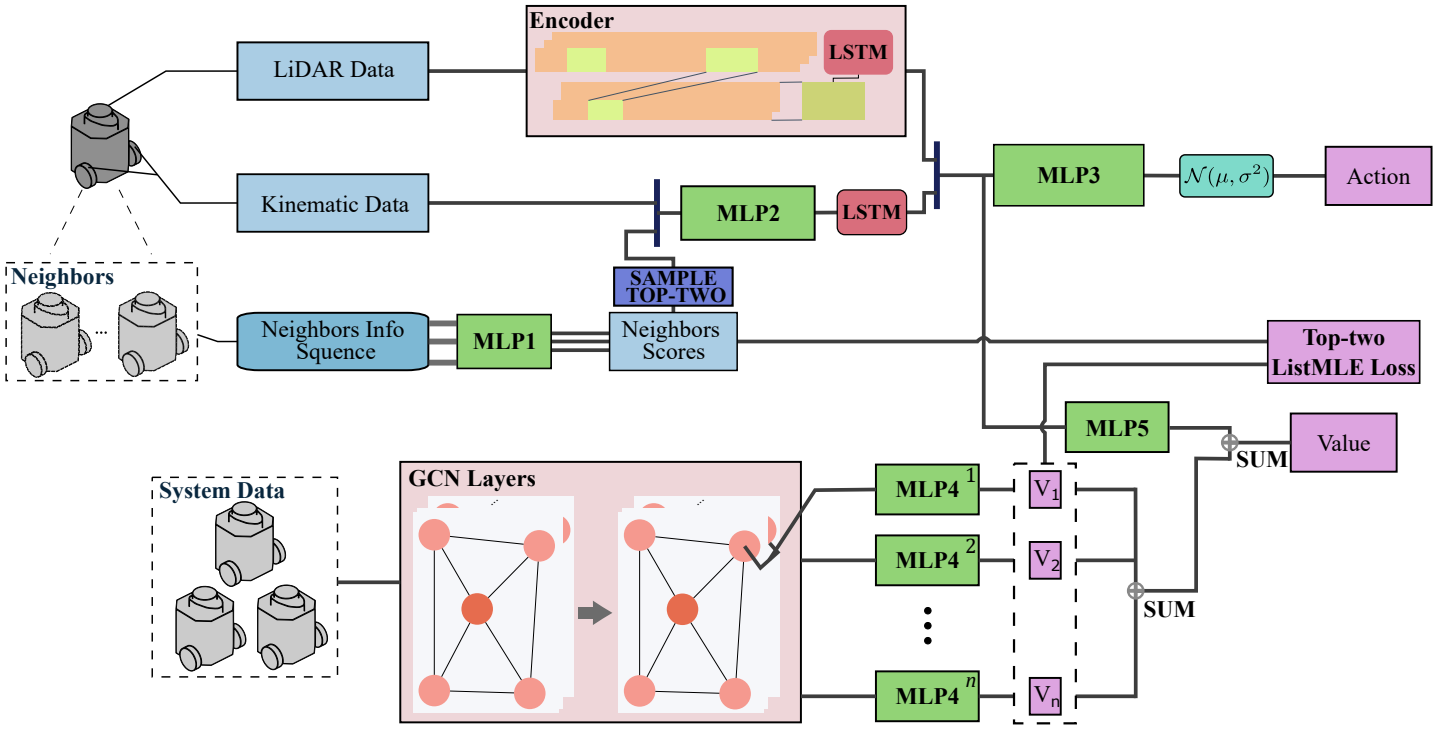


Fig. 2: The actor and critic network model employs a multiple-branch architecture to handle heterogeneous input sources. The encoder with one-dimensional convolutional layers handles LiDAR data through spatial feature extraction, while the GCN layers process system-level data by modeling relationships between data of each UGV in system.

the value estimation by the centralized critic for each UGV. The top two neighbors are not directly from neighbors of top-two score, but selected probabilistically based on their scores to enhance exploration performance. To train both the scoring network and the policy network, top-two ListMLE loss is added into PPO loss that

$$\mathcal{L} = \mathcal{L}_A + c_v \mathcal{L}_C + c_e \mathcal{L}_E + c_{\text{MLE}} \mathcal{L}_{\text{MLE}}, \tag{9}$$

where $\mathcal{L}_E$ is the policy entropy loss, c_v is value loss coefficient, c_e is the entropy coefficient, and c_{MLE} is the ListMLE coefficient.

B. Observation, Action and Reward

The environmental observations for each UGV comprise:

- Neighbor information sequence $\mathbf{q}_i$, where each entry contains relative position p_{ij}, relative velocity $\mathbf{v}_{ij}$, formation error $e_{ij}^f = \|p_{ij}\| - d_{ij}^*$, and leader indicator flag l_j for all neighbors $j \in \mathcal{N}_i$;
- LiDAR measurements $\mathbf{s}_i = [s_1^i, s_2^i, \cdots, s_m^i]^\top$ consisting of m sensor readings;
- Current velocity state $\mathbf{v}_i = [v_i, \omega_i]^\top$ detailing linear and angular components;
- Acceleration measurements $\mathbf{a}_i = [ax_i, a_{yi}]^\top$ in Cartesian coordinates.

A PI controller regulates wheel speed, while a reinforcement learning policy generates velocity commands to accomplish system control objectives. The action space for UGV i contains two primary control outputs:

- Linear velocity v_i command;
- Angular velocity ω_i command.

There are multiple objectives in our formation control with collision and collision avoiding, and each objective corresponds to a reward function, where R_{gf} is the global formation reward, R_{lf} the local formation reward, R_{Lf} the leader formation reward, R_o the collision avoiding reward, R_d is the dropping out avoiding reward, R_a the action smoothing reward, and R_v the global formation forming speed reward. The total reward R for policy training is a weighted sum of all rewards, where the weight coefficients are defined as follows

$$R = \alpha \begin{bmatrix} R_{gf} & R_{lf} & R_{Lf} & R_o & R_d & R_a & R_v \end{bmatrix}^\top, \tag{10}$$

where α is the coefficient vector. All the reward functions are specified below.

The global formation performance reward R_{gf} quantifies the overall formation accuracy using the system-level error term (4) as:

$$R_{gf} = -\ln\left(\varepsilon + \frac{e_f}{\kappa}\right). \tag{11}$$

Here, κ represents the spatial metric for the target formation configuration, calculated as $\kappa = \max d_{ij}^*$, while $\varepsilon > 0$ serves as a minimal stabilizing constant. Correspondingly, the neighborhood formation reward R_{fl} evaluates individual UGV i coordination with adjacent UGVs through:

$$R_{fl} = -\ln\left(\varepsilon + \frac{e_{if}}{\kappa}\right), \tag{12}$$

where e_{if} denotes the localized formation error specified in (3). The leader's formation reward R_{Lf} follows similar computation as (11), substituting e_f with leader's formation error $e_{fL} = \sum_{i=1}^{n} b_i \left| \|p_{i0}\| - d_{i0}^* \right|$.

Collision prevention compensation R_o measures other UGVs proximity by aggregating processed LiDAR measurements through transformation operator J_o:

$$R_o = -\sum J_o(\rho_i). \tag{13}$$

Utilizing a graded penalty function $J_o(\rho_i)$:

$$J_o(\rho) = \begin{cases} \frac{2(\rho - 2\rho_{\text{safe}})^2}{\rho_{safe}^2} & \rho \leq 2\rho_{\text{safe}} \\ 0 & \rho > 2\rho_{\text{safe}} \end{cases}, \tag{14}$$

where ρ_{safe} defines the critical distance threshold for collision risk assessment.

The connectivity preservation reward R_d penalizes excessive inter UGV separation to maintain formation integrity:

$$R_d = -\sum J_d(||p_{ij}||). \tag{15}$$

Utilizing another graded penalty function J_d:

$$J_d(\delta) = \begin{cases} 0 & \delta \leq \frac{1}{2}\delta_{\text{limit}} \\ \frac{4(\delta - \frac{1}{2}\delta_{\text{limit}})^2}{\delta_{\text{limit}}^2} & \frac{1}{2}\delta_{\text{limit}} < \delta \leq \delta_{\text{limit}} \\ 1 & \delta > \delta_{\text{critical}} \end{cases}, \tag{16}$$

where δ_{limit} specifies maximum permissible neighbor separation and δ_{critical} indicates communication failure distance.

Policy stability encouragement R_a implements temporal smoothing through:

$$R_a = -(a_t - a_{t-1})^2. \tag{17}$$

This term regulates controller output variations between consecutive time steps.

The formation error convergence speed rewards R_v is

$$R_v = \exp\left(\frac{\Delta e_f}{\zeta}\right), \tag{18}$$

where $\Delta e_f = e_f(k) - e_f(k-1)$ represents temporal formation error differential, with normalization factor $\zeta > 0$ moderating the progression rate sensitivity through exponential function.

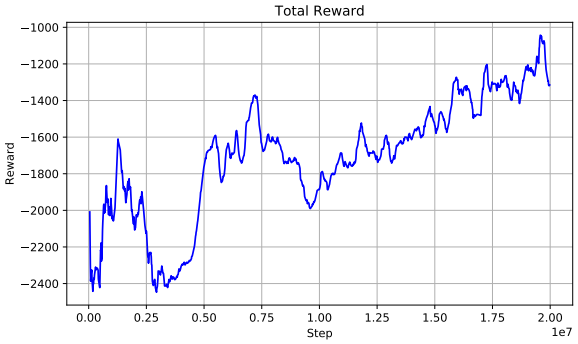
C. Training

In training, the observations between different UGVs and parallel environment are packed into batches. Length of neighbors sequence of each batch or UGV may be different, and processing these sequences with varying lengths iteratively can hinder training efficiency. To solve this problem, padding is added into the neighbors sequence forcing the lengths of all batched to be equal, and these padded sequences are then handled through vectorized processing to improve computational efficiency. However, the original ListMLE algorithm may not treat paddings, so we propose a method to handle the padding when use top-two ListMLE.

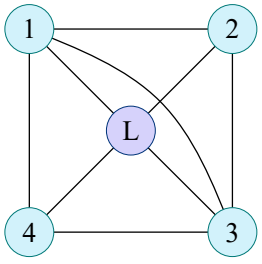
Specially, we add a tensor m to indicate the location of padding the input neighbors sequence tensor, where 0 indicates padding and 1 the normal data. The score from scoring model in the place indicated as padding are set to the value of $-I$, where I is a large enough value that $\exp(-I) \to 0$. So that the padding will not affect the top-two ListMLE loss when training. The padding aligns the neighbors sequence length to match the total number of UGVs, making vectorized processing of the neighbors sequence feasible.

IV. Experiment

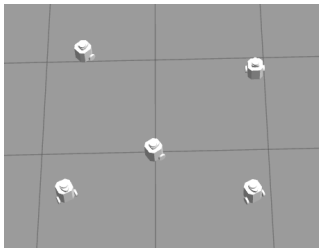
Policy training and evaluation were conducted within the Gazebo simulation platform featuring real-time physics engine. The experimental setup employs a turtlebot3 mobile platform - a differential drive unmanned UGV integrated with LiDAR sensors, utilizing Robot Operating System (ROS) for inter-module communication. During the learning phase, each training episode terminates upon either physical contact with other UGVs or reaching the maximum allowable operational period. Our implementation leverages the RLlib [10] reinforcement learning framework, with computational tasks executed on a workstation.



(a) The total reward curve during training.



(b) The desired formation pattern and communication topology.



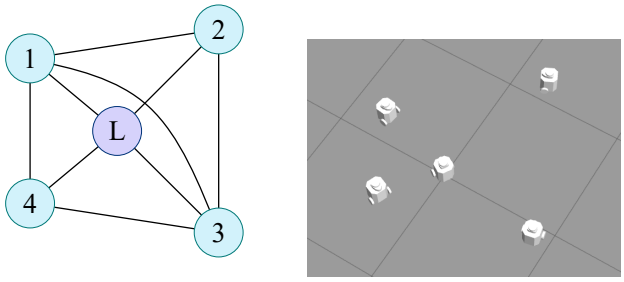
(c) The final formation formed.

Fig. 3: The first experiment to train a basic formation policy.

The primary control object involves coordinating unmanned ground vehicles to assemble into a predetermined static formation at target static coordinates. Initial conditions reset episodically, with follower UGVs dispersed arbitrarily across a sector at random orientations, while the leader is positioned at the geometrically central point of the intended formation pattern.

A policy is first train for basic formation control. The formation pattern targets a rectangular geometry spanning $1\mathrm{m}^2$ that one leader stabilizes at the centroid, while four followers align with the corners (visualized in Fig. 3b). The average time cost of the policy to generate an action is approximately 0.02s, which is 50Hz for control frequency. Training emphasizes foundational formation compliance with the result in Fig. 3c.

Next, the pattern of the target formation is modified by lower the distance between 1st, 4th and leader UGV as shown in Fig. 4a, then the pre-trained model is employed for formation control, with the results demonstrated in Fig. 4b. Compared with [6], since the formation data are not embedded in the critic network, the policy network does not require retraining when the formation configuration changes.



(a) The modified formation pattern.

(b) The final modified formation formed.

Fig. 4: The second experiment to show the ability to change formation pattern of pre-trained policy.

The results in Figs. 3-4 validate that the framework retains formation stability without policy retraining when transitioning from the original pattern to a modified configuration with different distance between UGVs.

V. Conclusion

This paper presents a reinforcement learning-based approach for achieving formation control with changeable pattern for UGVs. The proposed policy employs a top-two selection mechanism trained with ListMLE loss to acquire formation error data and other information from key neighbors, complemented by GCN layers for system value prediction. Experimental results confirm the feasibility of our method and its capability for formation pattern change. Future research directions include conducting real-world experimental validation with physical UGVs and investigating the interaction effects of multiple reward coefficients on policy behavior to achieve more precise formation control.

References

[1] M. Lewis and K. Tan, "High precision formation control of mobile robots using virtual structures," *Autonomous Robots*, vol. 4, no. 4, pp. 387–403, 1997.

[2] T. Balch and R. Arkin, "Behavior-based formation control for multi-robot teams," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 926–939, 1998.

[3] W. Zhao, H. Liu, F. Lewis, and X. Wang, "Data-driven optimal formation control for quadrotor team with unknown dynamics," *IEEE Transactions on Cybernetics*, vol. 52, no. 8, pp. 7889–7898, 2022.

[4] C. Bai, P. Yan, W. Pan, and J. Guo, "Learning-based multi-robot formation control with obstacle avoidance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11811–11822, 2022.

[5] Y. Zhao, Y. Ma, and S. Hu, "USV formation and path-following control via deep reinforcement learning with random braking," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 12, pp. 5468–5478, 2021.

[6] Y. Yan, X. Li, X. Qiu, J. Qiu, J. Wang, Y. Wang, and Y. Shen, "Relative distributed formation and obstacle avoidance with multi-agent reinforcement learning," in *2022 International Conference on Robotics and Automation (ICRA)*, (Philadelphia, PA, USA), pp. 1661–1667.

[7] Y. Zhou, F. Lu, G. Pu, X. Ma, R. Sun, H. Chen, and X. Li, "Adaptive leader-follower formation control and obstacle avoidance via deep reinforcement learning," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Macau, China), pp. 4273–4280.

[8] Y. Hu, J. Fu, and G. Wen, "Graph soft actor–critic reinforcement learning for large-scale distributed multirobot coordination," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2023.

[9] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li, "Listwise approach to learning to rank: theory and algorithm," in *Proceedings of the 25th international conference on Machine learning*, pp. 1192–1199, 2008.

[10] E. Liang, R. Liaw, R. Nishihara, P. Moritz, R. Fox, K. Goldberg, E. Joseph, I. Michael, and I. Stoica, "RLlib: Abstractions for distributed reinforcement learning," in *2018 International Conference on Machine Learning (ICML)*, (Stockholmsmässan, Stockholm SWEDEN).