2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

Dynamic Swarm Reconfiguration via Multi-Level Adaptive Cooperative Architecture for Multi-Robot Exploration

Shohei Inoue[1], Kosuke Sakamoto[2], and Yasuharu Kunii[3]

Abstract— **This study proposes a hierarchical cooperative control architecture for multi-robot exploration, centered on a dynamic branching–integration mechanism that adapts swarm structures to environmental conditions. The architecture integrates a *System Agent* for global coordination and a *Swarm Agent* for local action decisions, enabling adaptive and scalable cooperation.**

Simulations in multiple environments show that the branching–integration mechanism alone surpasses the baseline in all cases, with gains in complex settings. Learning-based optimization of branching–integration conditions further improves early-stage exploration speed and robustness.

These results indicate that dynamic swarm reconfiguration is the primary driver of exploration efficiency, while learning effectively enhances its benefits. Future work will investigate other reinforcement learning algorithms and exploration strategies, evaluate performance under dynamic or communication-constrained conditions, and conduct field tests with real robots.

I. Introduction

In recent years, the targets of planetary exploration have expanded beyond surface surveys to subsurface environments. In 2009, the scientific team of Japan's lunar orbiter SELENE (Kaguya) detected a lava tube in the Marius Hills region of the Moon. These tubes, presumed to have formed from ancient lava flows, may contain vast cave-like spaces underground [1]. Accessing such environments requires exploration methods more flexible and adaptive than conventional large-scale rovers.

Swarm robotics has emerged as a promising approach for autonomous exploration in unknown and unstructured environments such as disaster sites, underground spaces, and planetary surfaces. Unlike single high-performance robots, swarms of simple distributed robots offer robustness, scalability, and flexibility, enabling effective operation under poor communication and high uncertainty. Recent advances include probabilistic density control methods and the Adaptive Flocking Algorithm [2][3].

Incorporating reinforcement learning into swarm robotics further enables efficient and adaptive exploration in environments where rule-based models struggle [4]. Robot swarms can autonomously optimize their policies based on environmental conditions and peer movements, achieving cooperative behaviors such as task allocation, path planning, and dynamic obstacle avoidance.

[1]Shohei Inoue is a master's student at Chuo University, Tokyo, Japan. a20.mskr@g.chuo-u.ac.jp

[2]Kosuke Sakamoto is an Associate Professor at the same department, Chuo University. ksakamoto605@g.chuo-u.ac.jp

[3]Yasuharu Kunii is a Professor at the same department, Chuo University. kunii@elect.chuo-u.ac.jp

While research on local cooperation and self-organization has progressed, higher-level organizational control strategically altering swarm structures according to environmental and task conditions, including dynamic formation and dissolution remains underexplored.

We propose a **multi-level adaptive cooperative architecture** in which a swarm of small exploration robots autonomously repeats branching and merging to efficiently explore wide areas in unknown environments, by separating learning targets into "system-level strategy" and "swarm-level behavior."

In conventional single-layer strategies, global decision-making and individual behavior control are intertwined, making it difficult to switch to optimal swarm structures in response to sudden environmental changes or exploration progress. Relying solely on local information risks biased coverage and high revisit rates, while focusing on global optimization reduces responsiveness to local conditions, creating a trade-off.

In the proposed method, the upper-level *System Agent* monitors overall exploration progress and swarm distribution, dynamically optimizing branching and merging to allocate exploration resources spatially. The lower-level *Swarm Agent* makes immediate decisions from local observations, handling obstacle avoidance and improving local exploration efficiency.

This hierarchical strategy enables swarm robots to maintain both global exploration efficiency and local adaptability, preserving scalability and robustness under unknown and dynamically changing environmental and communication conditions.

The verification is conducted using a simulation environment for small swarm robots intended for planetary and underground exploration. Fig. 1 shows the assumed robot model. As described in Chapter III, this robot is small and, due to physical and capability constraints, has low individual intelligence, aiming to achieve environmental adaptability through cooperative behavior among multiple units.

Fig. 1: Small robot model assumed in this study

The structure of this paper is as follows:

- Chapter II: Review of related work on swarm coordination control, swarm-based exploration, and deep reinforcement learning (DRL).
- Chapter III: Problem setting and the proposed multi-level adaptive cooperative architecture.
- Chapter IV: Simulation study, including experimental settings, learning algorithm configurations, and results under different environmental and learning conditions.
- Chapter V: Conclusions and future research directions.

II. Related Work

This section reviews related studies on swarm coordination, swarm robot exploration, and DRL.

A. Cooperative Behavior of Swarm Robots

Research on swarm coordination originates from models inspired by the behavioral patterns of biological swarms, with the Boids model and the Vicsek model being representative examples [5], [6]. In these models, each individual follows simple local rules, leading to the spontaneous formation of organized patterns at the group level. Such cooperative behavior plays an important role in swarm robot control, as it enables scalable and robust swarm formation without relying on centralized control.

B. Exploration by Swarm Robots

Exploration by swarm robots has attracted significant attention as an important method for achieving efficient area coverage in unknown environments [7]. By deploying a large number of robots in a distributed manner, it is possible to explore a wide range of the environment while ensuring robustness and scalability. Moreover, probabilistic local control schemes with collision avoidance in mind (e.g., probabilistic VFH) have been reported to be effective in improving reliability and path planning performance for swarm robot exploration in unknown environments [8]. Existing studies have proposed various algorithms aimed at minimizing path overlap and maximizing exploration range; however, many of these approaches are limited to simple distributed behaviors or leader dependent control [9], [10].

C. Deep Reinforcement Learning in Robotics

DRL has brought significant advancements to robot control by enabling end-to-end learning from sensor inputs to actions [11], [12]. It has been successfully applied to a variety of robotic tasks, such as navigation, manipulation, and exploration. In particular, its ability to adapt to environments without relying on pre defined models makes it well suited for autonomous exploration in unknown environments [13], [14]. In the context of swarm robotics, DRL has been applied to learning collective behaviors such as formation control, target tracking, and cooperative exploration, and has evolved into distributed reinforcement learning and multi agent DRL [15], [16]. However, most existing studies assume simple cooperative structures or homogeneous populations, and have yet to sufficiently address complex organizational formations or heterogeneous cooperation.

To address these challenges, Hierarchical Reinforcement Learning (HRL) has attracted attention as a method that separates long term goals from short term actions, enabling more abstract and structured decision making [17]. Representative approaches include the Options Framework and FeUdal Networks (FuN), both of which have demonstrated improvements in exploration efficiency and learning stability through task decomposition [18], [19].

In recent years, the importance of dynamically adapting swarm structures according to environmental conditions has been highlighted in the field of unknown environment exploration by swarm robots. However, most existing approaches focus either on low level coordination or high level planning, and the autonomous reorganization of swarm strategies has not been sufficiently explored. To address this challenge, in this study we apply the hierarchical approach (HRL) to swarm robot exploration, separating strategic decision making at the system level from behavioral control at the swarm level, enabling branching and merging of swarms according to environmental conditions. We propose a **multi level adaptive cooperative architecture** that enables dynamic swarm reconfiguration. Furthermore, a learning module is introduced as an extended element to optimize the branching and merging strategies within this architecture.

III. Proposed Method

A. Overall Structure of the Proposed Architecture

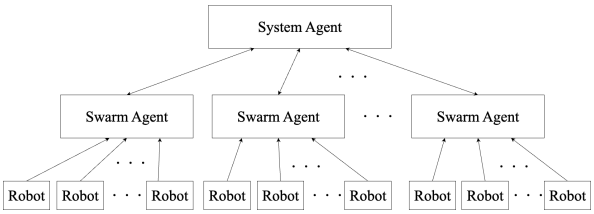


Fig. 2: Overview of the multi level adaptive cooperative architecture

This study proposes a multi-level adaptive cooperative architecture centered on a swarm branching–integration mechanism. The architecture consists of the following three layers:

- **System Level**: Decision making for swarm branching and merging
- **Swarm Level**: Determination of swarm movement direction based on traversability and exploration efficiency
- **Individual Level**: Random-walk based exploration behavior

Hierarchical reinforcement learning can be applied as needed to optimize branching–integration strategies and behavioral parameters. The following subsections describe each level in detail.

B. Individual-Level Control (Robot Constraints and Capabilities)

The robots considered in this study are small swarm robots designed for planetary exploration and underground environments, with the following physical and sensing limitations:

- **Size and mobility**: Small wheeled form factor, capable only of straight line movement and in place rotation.
- **Sensing capability**: Equipped with an infrared distance sensor capable of detecting obstacles within 15 cm in the forward direction. In addition, Ultra-Wideband (UWB) wireless communication is used to estimate the relative position and orientation of the exploration center (leader). No environmental mapping function or long range sensors are installed.
- **Self-localization**: Using UWB based positioning, each robot can estimate its relative position and orientation within the swarm coordinate frame. Absolute positioning (coordinates in a global map) is unavailable.
- **Local cooperation**: Based on relative position information from the leader robot, each unit follows the leader and maintains swarm cohesion. High level path planning and exploration strategy decisions are delegated to upper layers.

Due to these constraints, it is difficult for a single robot to make advanced exploration decisions independently. Therefore, in this work, each unit is dedicated to local cooperative behavior, while upper levels (swarm level and system level) handle strategic decision-making in a hierarchical control framework.

C. Swarm-Level Control (Swarm Formation and Movement Direction)

The swarm is organized through *Probabilistic Density Control* (PDC). In PDC, each robot observes local density and moves toward lower-density directions using MCMC based accept–reject sampling, thereby maintaining both swarm cohesion and uniform spatial distribution. This approach suppresses congestion in high-density regions and ensures robust behavior against local minima and environmental changes.

In this study, a PDC swarm composed of one leader and multiple follower robots is guided by a probabilistic navigation method combining the existing *Vector Field Histogram* (VFH) algorithm with fuzzy inference for movement direction selection. The angular space is divided into N segments, and for each direction θ_i, traversability P_d and exploration improvement P_e are computed. P_d is calculated from obstacle information as a traversability histogram D, while P_e is evaluated using a von Mises distribution to suppress movement in the reverse of the previous direction. The two are integrated using a soft threshold fuzzy inference function to obtain the final histogram R, from which the movement direction $\theta_{selected}$ is chosen via quartile-based weighted probabilistic sampling.

This configuration enables each robot to probabilistically select a safe and efficient movement direction based on local obstacle information and exploration history.

D. System-Level Control (Branching–Merging Strategy)

At the system level, decisions are made regarding swarm branching and merging. Each robot computes a *mobility score* s_m for its exploration sector under PDC, defined as:

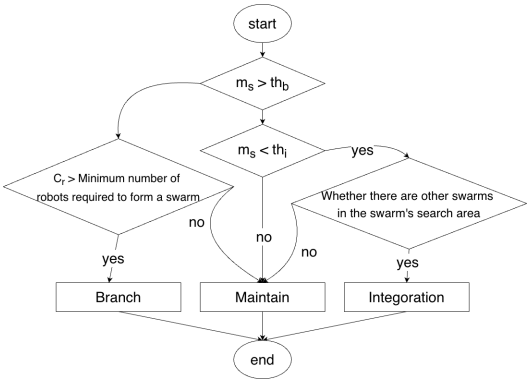


Fig. 3: Flowchart of branching and merging decisions

$$s_m = \alpha \cdot c_i + \beta \cdot m_i + \gamma \cdot e_i \tag{1}$$

c_i is the proportion of collisions in exploration sector i, m_i is a comparative metric between internally computed movement and externally observed movement from sensing data, and e_i is the exploration contribution calculated from m_i. The parameters α, β, and γ are weighting coefficients.

Since a swarm is a collective of robots, branching and merging can be achieved without direct external observation by using both individual robot data and aggregated swarm information. Additionally, the number of robots in the swarm c_r is compared with pre defined thresholds for branching th_b and merging th_i to determine structural changes.

Implementing branching–integration enables dynamic adaptation to environmental conditions, improving exploration efficiency. Moreover, branching allows the formation of multiple swarms, enabling task specialization and the development of new organizational capabilities. Fig. 3 shows the decision flow for branching and merging.

E. Key Features of the Proposed Architecture

The main features of the proposed architecture are as follows:

- **Flexibility via hierarchical structure:** Functions can be added or modified at each level, and each level can be optimized or adjusted independently. By clearly separating concerns, global strategy and local adaptation remain decoupled, allowing each level to focus on its specific responsibilities without interference.
- **Enhanced adaptability:** Dynamic branching and merging in response to environmental changes make the approach well suited for complex multi robot exploration scenarios. It retains the scalability and robustness inherent to swarm systems while addressing the complexity of multi agent cooperation.
- **Comparison with conventional swarm control:** Traditional swarm control often involves single-layer control or fixed swarm structures. The proposed approach enables dynamic restructuring through a hierarchical framework, greatly improving environmental adaptability.

Furthermore, integrating a learning mechanism allows autonomous strategy decisions and flexible behavior adaptation to varying environments.

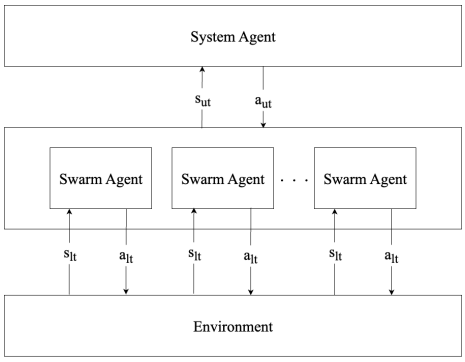


Fig. 4: Learning configuration of the architecture

The overall learning configuration is shown in Fig. 4. The system can be formulated as hierarchical reinforcement learning, consisting of a *system-level policy* π_s and a *swarm-level policy* π_w.

The hierarchical policy is defined as:

$$\pi_h = (\pi_s, \pi_w) \tag{2}$$

$$\pi_s : s_{ut} \rightarrow a_{ut} \quad \text{(System-level policy)} \tag{3}$$

$$\pi_w : s_{lt} \times a_{ut} \rightarrow a_{lt} \quad \text{(Swarm-level policy)} \tag{4}$$

s_{ut} denotes the system agent's state space and a_{ut} its action space (branch, integration, maintain). s_{lt} is the swarm agent's state space, and a_{lt} is its action space (movement direction, cooperation parameters).

The hierarchical value function is expressed as:

$$V_h(s) = V_s(s_{ut}) + \mathbb{E}_{\pi_w}[V_w(s_{lt}, a_{ut})] \tag{5}$$

Where $V_s(s_{ut})$ is the system level value function, and $V_w(s_{lt}, a_{ut})$ is the swarm level value function conditioned on a_{ut}.

The overall objective function is:

$$J_h(\theta_s, \theta_w) = \mathbb{E}_{\pi_h}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{6}$$

Where θ_s and θ_w are the parameters of the system and swarm policies, respectively, and r_t is the reward at time t.

System Level: The system agent policy is defined as:

$$\pi_s : s_{ut} \rightarrow a_{ut}, \quad a_{ut} \in \{\text{Branch}, \text{Integration}, \text{Maintain}\} \tag{7}$$

It determines dynamic swarm restructuring based on overall exploration progress, swarm configuration, and performance metrics. The reward is designed to evaluate coverage improvement and adaptive swarm management.

Swarm Level: The swarm agent policy is defined as:

$$\pi_w : s_{lt} \times a_{ut} \rightarrow a_{lt} \tag{8}$$

It determines local movement actions based on the environment and upper level commands. In particular, it learns three parameters related to direction selection with VFH and fuzzy inference (safety threshold th, exploration improvement k_e, and collision suppression k_c) to optimize the balance between local exploration efficiency and collision avoidance.

This hierarchical structure enables integrated global policy and local control, allowing adaptive branching–integration and cooperative exploration in response to environmental changes.

IV. Simulation Study

A. Experimental Setup

(a) No Obs (b) Sparse Obs (c) Dense Obs

Fig. 5: Environments used in the experiments

The primary objective of this study is to evaluate the impact of the proposed architecture's branch–integration mechanism on exploration performance. In addition, we assess the potential performance improvements achieved by incorporating a learning module. To this end, we compare the following three approaches in the three environments shown in Fig. 5.

- **A**: Baseline (single swarm exploration)
- **B**: Baseline + Proposed framework
- **C**: Baseline + Proposed framework + A2C

Method A (Baseline) employs a single swarm of 20 robots exploring the environment using the VFH-Fuzzy algorithm with fixed default parameters (safety threshold $th = 0.5$, exploration improvement $k_e = 10.0$, collision suppression $k_c = 5.0$) for movement direction selection. The System Agent's branching and integration mechanisms are disabled, and no learning is applied. This configuration represents a basic swarm exploration approach without dynamic reconfiguration or adaptive parameter optimization.

The number of robots was fixed at 20. For Method C, we adopted the Advantage Actor–Critic (A2C) algorithm as an initial investigation. The System Agent learns the threshold parameters for branching and merging based on statistical information from the Swarm Agent, while the Swarm Agent learns the parameters used for movement direction decisions (th, k_e, k_c).

The following evaluation metrics are used to assess the performance of each method.

Exploration Rate: The exploration rate $R_{exp}(t)$ at time step t is defined as:

$$R_{exp}(t) = \frac{N_{explored}(t)}{N_{total}} \tag{9}$$

where $N_{explored}(t)$ is the number of explored cells at time step t, and N_{total} is the total number of explorable cells in the environment (excluding obstacle cells).

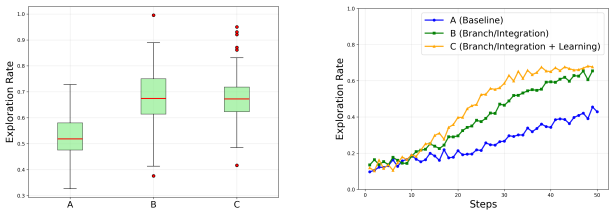
Exploration Efficiency: The exploration efficiency E_{exp} is defined as the exploration rate achieved per unit step:

$$E_{exp} = \frac{R_{exp}(T)}{T} \tag{10}$$

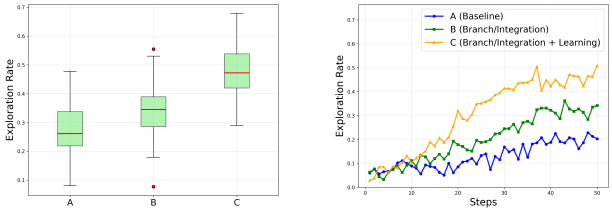
where T is the total number of steps taken. Additionally, we evaluate the time-to-target, defined as the number of steps required to achieve a specific exploration rate threshold (e.g., 50% or 80%).

The exploration duration was set so that the baseline method A achieved a coverage rate of approximately 50–60% in Environment A.

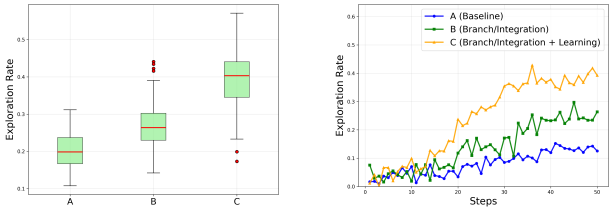
B. Experimental Results and Discussion



(a) Environment A



(b) Environment B



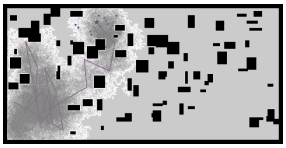
(c) Environment C

Fig. 6: Simulation results: Distribution of final exploration rates and exploration rate progression over episodes for each environment.

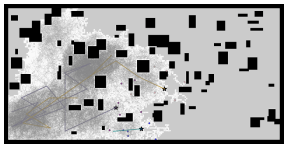
Fig. 6 presents the experimental results. From the top row of Fig. 6, Method B (proposed architecture with branch–integration) outperformed the baseline A in all environments. In particular, in Environments B and C, the final exploration rate improved by +9.8% and +16.2%, respectively, compared to the baseline. This effect became more pronounced as environmental complexity increased.

Furthermore, adding the learning module (Method C) provided additional improvements over Method B, achieving improvements of +11.5% in Environment B and +16.2% in Environment C compared to the baseline. This suggests that learning-based optimization of conditions is effective in more complex environments.

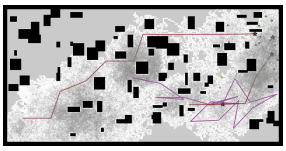
In the top row of Fig. 6, both B and C have higher medians and smaller interquartile ranges (IQR) than A. For example, in Environment A, Method B achieved the smallest IQR of 1.8%, while in Environment C, Method C had the most stable



(a) Baseline



(b) Proposed architecture



(c) Proposed architecture + Learning

Fig. 7: Final exploration status in Environment C. White areas represent explored regions, black areas are obstacles, thick lines indicate leader robot trajectories, thin lines indicate follower robot trajectories, dots mark follower robot positions, and stars mark leader robot positions.

IQR of 2.1%. This indicates reduced variability and fewer outliers, reflecting improved robustness.

From the temporal progression in the bottom row of Fig. 6, both B and C achieved higher exploration rates earlier than A, with C showing an even steeper initial growth rate. However, Method B alone already demonstrated substantial early growth, indicating that the primary driver of improved exploration efficiency is the branch–integration mechanism.

From the perspective of exploration efficiency, Methods B and C also outperformed baseline A in terms of exploration rate improvement per unit step. In particular, in Environment C, Method C achieved an improvement of approximately 15% or more in average exploration efficiency compared to the baseline, and reduced the time to reach 80% exploration rate. This indicates that dynamic swarm reconfiguration and adaptive parameter adjustment through learning contribute to improved exploration speed.

1) Effect of Branch–Integration Mechanism: Results from Method B show that introducing only the branch–integration mechanism outperformed the baseline in all environments. This demonstrates that dynamic adaptation of swarm structure contributes to improved exploration efficiency, with the effect becoming more significant in complex environments.

2) Additional Effect of Learning Module: Method C exhibited further improvements over Method B. In highly complex environments, learning adaptively adjusted the timing and conditions for branching and integration, increasing the growth rate from the early stages of exploration. However, the main factor in improving exploration efficiency remained the branch–integration mechanism, with learning serving a supplementary role.

3) Environmental Dependency: While Method B achieved the highest average exploration rate (96.8%) in Environment A, Method C showed additional benefits in Environments B and C. In simple environments with low obstacle density and less complex terrain, rule-based branching and integration provided sufficient performance. In contrast,

learning-based optimization proved effective in more complex environments. As a representative case, Fig. 7 illustrates the final exploration status in Environment C, where the proposed architecture (B and C) covers a broader area than the baseline (A).

4) Robustness Improvement: The reduction in variability of average exploration rates and the decrease in outliers demonstrate that the proposed method maintains stable exploration performance under diverse conditions. In particular, Method B ensured overall stability, while Method C further reduced failure cases.

V. Conclusion

In this study, we proposed a hierarchical cooperative control architecture for multi-robot exploration tasks, centered on the swarm division and integration function. The proposed architecture integrates global coordination by the System Agent with local action decision-making by the Swarm Agent, enabling dynamic swarm reconfiguration according to environmental conditions.

Simulation results demonstrated that Method B, which incorporated only the division and integration function, outperformed the baseline in all environments and significantly improved exploration efficiency, particularly in complex environments. Furthermore, Method C, which added a learning mechanism, achieved additional performance gains by optimizing the timing and conditions of division and integration, building upon the benefits of the basic function.

These results indicate that the primary factor in improving exploration performance lies in the architecture's capability for dynamic swarm reconfiguration, while learning serves as an effective auxiliary means to enhance this effect.

Future work will focus on integrating other reinforcement learning algorithms and exploration strategies, adapting to newly introduced functions at each level, and conducting field tests with real robots to further assess applicability in real-world environments.

A. Limitations and Future Work

While the findings of this study are promising as an initial investigation, several limitations and future research directions have been identified.

- **Expansion to diverse exploration strategies**: It is necessary to examine the integration of strategies beyond division and integration.
- **Adaptation to constraints for real robot deployment**: There is a need to develop lightweight and distributed processing methods that account for communication delays and limited computational resources.

Future research directions include:

1) Designing distributed reinforcement learning algorithms that consider intra-swarm communication constraints.
2) Demonstrating the algorithm and tuning parameters through field experiments with real robots.
3) Comparing performance and expanding capabilities through integration with other cooperative strategies.

ACKNOWLEDGMENT

This work was supported by JST, Moonshot Research and Development Program, grant number JPMJMS2238, Japan, by JSPS KAKENHI Grant Number JP24K17245, and by the Chuo University Joint Research Grant.

References

[1] T. Kaku *et al.*, "Detection of intact lava tubes at Marius Hills on the Moon by SELENE (Kaguya) Lunar Radar Sounder," *Geophysical Research Letters*, vol. 44, pp. 10,155–10,161, 2017.

[2] T. Sato et al., "Probabilistic Presence Density Control for Areal Wide-area Distributed Exploration with Swarm Robots", Journal of the Robotics Society of Japan, vol. 41, no. 10, pp. 869–880, 2023.

[3] Lee Geunho, Chong Nak Young, "Adaptive Flocking of Robot Swarms: Algorithms and Properties", IEICE Transactions on Communications, vol. E91-B, no. 9, pp. 2848–2855, 2008.

[4] J. Kober et al., "Reinforcement learning in robotics: A survey", The International Journal of Robotics Research, vol. 32, no. 11, pp. 1238–1274, 2013.

[5] Reynolds, C. W., "Flocks, Herds and Schools: A Distributed Behavioral Model", ACM SIGGRAPH Computer Graphics, vol. 21, no. 4, pp. 25–34, 1987.

[6] Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., and Shochet, O., "Novel Type of Phase Transition in a System of Self-Driven Particles", Physical Review Letters, vol. 75, no. 6, pp. 1226–1229, 1995.

[7] Muhsen, D. K., Sadiq, A. T., & Raheem, F. A. R., "A Survey on Swarm Robotics for Area Coverage Problem", Algorithms, vol. 17, no. 1, pp. 3 (2024).

[8] Sakamoto, Kosuke and Kunii, Yasuharu, "Probabilistic VFH-based Obstacle Avoidance Algorithm for Unknown Environment Exploration using Swarm Robots", IEEE/SICE International Symposium on System Integration (SII), 2025.

[9] Henard, A., Riviere, J., & Peillard, E., "A Self-Organizing Area Coverage Method for Swarm Robots Based on Gradient and Grouping", Symmetry (MDPI), vol. 13, no. 4, Article 680 (2021).

[10] Tran, V. P., Garratt, M. A., Kasmarik, K., & Anavatti, S. G., "Frontier-led Swarming: Robust Multi-Robot Coverage of Unknown Environments", arXiv preprint arXiv:2111.14295 (2021).

[11] Mnih, Volodymyr and Kavukcuoglu, Koray and Silver, David and et al., "Human-level control through deep reinforcement learning", Nature, vol. 518, no. 7540, pp. 529–533, 2015.

[12] Levine, Sergey and Finn, Chelsea and Darrell, Trevor and Abbeel, Pieter, "End-to-end training of deep visuomotor policies", Journal of Machine Learning Research, vol. 17, no. 39, pp. 1–40, 2016.

[13] Kober, Jens and Bagnell, J Andrew and Peters, Jan, "Reinforcement learning in robotics: A survey", International Journal of Robotics Research, vol. 32, no. 11, pp. 1238–1274, 2013.

[14] Kumra, Sulabh and Joshi, Saurabh and Saxena, Ashutosh, "Composing Robot Skills for Multi-Modal Room-to-Room Navigation", ICRA, 2019.

[15] Hüttenrauch, Max and Adrian, Adrian and Neumann, Gerhard, "Guided deep reinforcement learning for swarm systems", AAMAS, 2017.

[16] Jiang, Jiechuan and Lu, Zongqing, "Multi-agent deep reinforcement learning: A survey", arXiv preprint arXiv:2006.07889 (2020).

[17] Barto, Andrew G and Mahadevan, Sridhar, "Recent advances in hierarchical reinforcement learning", Discrete Event Dynamic Systems, vol. 13, no. 1, pp. 41–77, 2003.

[18] Sutton, Richard S and Precup, Doina and Singh, Satinder, "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning", Artificial Intelligence, vol. 112, no. 1-2, pp. 181–211, 1999.

[19] Vezhnevets, Alexander and Osindero, Simon and Schaul, Tom and et al., "FeUdal Networks for Hierarchical Reinforcement Learning", ICML, 2017.