# A reinforcement learning-based approach to cooperative multi-UAV task allocation

Naohiro Kubota*, Hideyoshi Miura†, Tomotaka Kimura‡, Kouji Hirata§
* Graduate School of Science and Engineering, Kansai University, Japan
† Graduate School of Interdisciplinary Science and Engineering in Health Systems, Okayama University, Japan
‡ Faculty of Science and Engineering, Doshisha University, Japan
§ Faculty of Engineering Science, Kansai University, Japan
E-mail: {k966930, hirata}@kansai-u.ac.jp, h-miura@okayama-u.ac.jp, tomkimur@mail.doshisha.ac.jp

***Abstract*—In this paper, we propose a reinforcement learning-based trajectory design method for multiple unmanned aerial vehicles (UAVs) for cooperative task allocation. Each UAV employs a Deep Q-Network (DQN) to process tasks in designated areas by traversing them as checkpoints. The UAVs jointly learn the optimal checkpoint assignments and efficient routes during the training process of the DQN. By sharing positional information to prevent collisions of UAVs, they cooperatively construct the shortest paths through their respective checkpoints, enabling flexible and safe autonomous flight that overcomes the limitations of single-agent operations. The effectiveness of the proposed method for cooperative task allocation of UAVs is demonstrated through simulation experiments.**

## I. Introduction

Unmanned Aerial Vehicle (UAV)-based network systems have gained increasing attention in various fields, including disaster response, precision agriculture, environmental monitoring, logistics, and communication support [1], [2], [3]. Their high mobility and flexibility enable efficient real-time data collection and communication over wide areas without relying on ground infrastructure [4], [5]. As missions in UAV-based network systems become more complex and large-scale, the need for cooperative operation among multiple UAVs has grown [6], [7]. Coordinated behavior, where UAVs autonomously divide tasks and adapt to environmental changes, is essential for mission success under dynamic and unpredictable situations.

Existing works have mainly focused on static task planning based on pre-determined routes [1], [2]. However, real-world environments are prone to sudden changes, such as communication disruptions and unforeseen obstacles. These situations require UAVs to make decentralized, real-time decisions that balance collision avoidance of UAVs with efficient task execution [6], [7]. To meet these demands, deep reinforcement learning has emerged as a promising approach. Specifically, the use of a Deep Q-Network (DQN) enables individual UAVs to learn optimal behaviors through interaction with the environment while maintaining decentralized cooperation [3], [5]. However, several challenges remain, including dynamic task allocation, balanced workload distribution, and responsive collision avoidance of UAVs.

This paper proposes a DQN-based decentralized trajectory design method for cooperative multi-UAV task allocation. In this method, using the DQN, UAVs share positional data with each other to autonomously determine their routes without collisions of UAVs. The routes pass through designated areas where tasks to be processed exist named task point, as shown in Fig. 1. The task points are autonomously assigned to one of the UAVs through the learning process of the DQN. We demonstrate the effectiveness of the proposed method for cooperative task allocation of UAVs through simulation experiments.

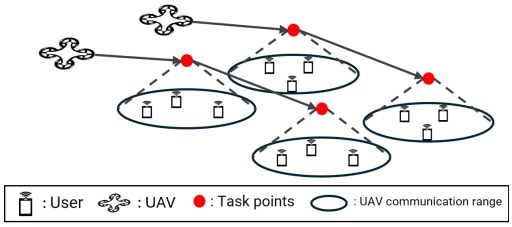


**Fig. 1:** Cooperative multi-UAV task allocation.

## II. System model

In this paper, we consider a system model in which $N$ identical UAVs operate in a closed two-dimensional area consisting of cells, as shown in Fig. 2. This paper does not consider the flight altitude of UAVs. Each UAV has its own starting point within the area, while a single common destination is shared by all UAVs. Furthermore, there exist some task points randomly placed within the area. We assume that there are no obstacles in the area. The UAVs are not allowed to exit the area.

We assume discrete time steps in the system model. Each UAV can move to one of six adjacent cells, i.e., upper-right, right, lower-right, lower-left, left, and upper-left, per time step. Each UAV can observe its own current position (i.e., current cell), the positions of the other UAVs, and the location of the next task point to be visited. Note that a task point is considered "visited" when a UAV reaches its cell. At each time step, the task point nearest to the UAV is selected as the next point to be visited. Based on these observations, each UAV learns an individual policy to act autonomously. Specifically, each UAV departs from its designated starting point and aims to reach the goal while passing through task points. In this process, each task point must be traversed by exactly one UAV.

979-8-3315-7206-8/25/$31.00 

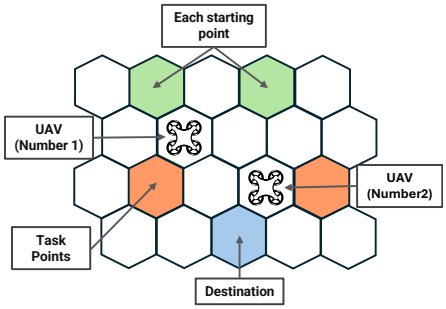


**Fig. 2:** System model

Under this constraint, the proposed method aims to determine the shortest routes of UAVs based on the DQN.

## III. Proposed Method

The proposed method enables cooperative UAV path planning by optimizing both task assignment and trajectory planning using the DQN. In this system, task assignment, i.e., determining which UAV should visit each task point, is not predefined by any rule. Instead, this is determined through reinforcement learning.

### A. Q-value

The proposed method defines the following state $s_t$, action $a_t$, and reward $r_t$ at time $t$ for the DQN.

1) **State:** As mentioned in the system model, each UAV can observe its own current position, the positions of the other UAVs, and the location of the next task point to be visited. Accordingly, we define the combination of these observations as the state $s_t$. Note that the visiting status of task points is represented by a binary vector.
2) **Action:** The action $a_t$ is the movement of the UAVs. Specifically, each UAV selects one of six adjacent cells and then moves to the selected cell. Note that the UAVs are not allowed to exit the closed area.
3) **Reward:** Table I shows the reward $r_t$ according to situations. A small negative reward is applied to every movement step to prevent unnecessary travel. A high reward is assigned to the UAV that first visits a task point. Furthermore, a reward is given for approaching an unvisited task point, where $d$ denotes the distance from the current cell to the task point. A higher reward is given when a UAV reaches the destination. Negative rewards are imposed for invalid movements, such as leaving the area or collision with other UAVs.

In the DQN, the action-value function $Q(s_t, a_t)$ named Q-value is used to determine the action policy. Given the state $s_t$ and action $a_t$ at time step $t$, the Q-value is updated as follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[ r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right], \tag{1}$$

where $\alpha$ denotes the learning rate and $\gamma$ denotes the discount factor. This update rule enables the agent to iteratively refine its Q-values, thereby approximating the optimal policy.

**TABLE I:** Reward.

| Situation | Reward |
|---|---|
| Step movement | $-2$ |
| First visit to a task point | $+20$ |
| Approaching unvisited task point | $+5/d$ |
| Reaching the destination | $+30$ |
| Leaving the area | $-10$ |
| Collision with other UAV | $-10$ |

### B. Action selection

In reinforcement learning, it is essential to maintain a balance between exploration, which enhances the agent's understanding of the environment, and exploitation, which leverages acquired knowledge to maximize cumulative rewards. To achieve this balance, the following two action selection strategies are employed in this paper:

1) **$\varepsilon$-Greedy**: In this strategy, an action is selected randomly with a probability of $\varepsilon$ (exploration), while the action with the highest Q-value for the current state $s_t$ is chosen with a probability of $1 - \varepsilon$ (exploitation). This action selection strategy can be expressed as follows:

$$a_t = \begin{cases} \text{random action}, & \text{with probability } \varepsilon, \\ \arg\max_a Q(s_t, a), & \text{with probability } 1 - \varepsilon. \end{cases} \tag{2}$$

In the proposed method, the exploration rate $\varepsilon$ is gradually decreased during training in order to shift the agents' behavior from exploration to exploitation. Specifically, at the end of each episode, the following updates are performed: $\varepsilon \leftarrow 0.999 \times \varepsilon$, with a minimum value of 0.1.
2) **Softmax**: In this strategy, actions are selected probabilistically according to their Q-values in the current state $s_t$. The selection probability for each action $a$ is computed using the Softmax function as follows:

$$P(a|s_t) = \frac{\exp(Q(s_t, a)/\tau)}{\sum_b \exp(Q(s_t, b)/\tau).} \tag{3}$$

where $\tau$ is a parameter that controls the degree of exploration. Specifically, a higher value of $\tau$ promotes exploration, whereas a lower value encourages exploitation. Similar to $\varepsilon$, at the end of each episode, the following updates are performed: $\tau \leftarrow 0.999 \times \tau$, with a minimum value of 0.1.

### C. Implementation of action selection strategy

In the proposed method, to dynamically balance exploration and exploitation in reinforcement learning, the entire training process is divided into five phases according to the total number of learning iterations. In each phase, a weighted combination of action selection strategies is employed.

The action selection strategies employed in this paper are the $\varepsilon$-greedy method, the Softmax method, and the Greedy method. The Greedy method always selects the action with the highest Q-value. In each phase of the training process, these

**TABLE II:** Training phases and strategy selection probability.

| Phase | Episode Range | $\varepsilon$-greedy | Softmax | Greedy |
|---|---|---|---|---|
| Initial | 0–20% | 0.90 | 0.10 | 0.00 |
| Mid-phase 1 | 20–40% | 0.70 | 0.30 | 0.00 |
| Mid-phase 2 | 40–60% | 0.40 | 0.50 | 0.10 |
| Late-phase 1 | 60–85% | 0.20 | 0.40 | 0.40 |
| Late-phase 2 | 85–100% | 0.05 | 0.30 | 0.65 |

three strategies are selected according to the probability listed in Table II. The rationale behind the strategy composition in each phase and its impact on the learning process are described below.

- **Initial Phase (0–20%):** The $\varepsilon$-greedy method is primarily employed to promote broad exploration and facilitate the initialization of Q-values by enabling visits to unexplored regions.
- **Mid Phase 1 (20–40%):** The Softmax method is introduced to allow value-based probabilistic action selection, thereby balancing extensive exploration with improved learning efficiency.
- **Mid Phase 2 (40–60%):** The Softmax method remains the primary strategy, with the partial incorporation of the Greedy method. This phase aims to refine the learned policy through a balanced approach to exploration and exploitation.
- **Late Phase 1 (60–85%):** The Softmax and Greedy methods are used in approximately equal proportions to exploit accumulated knowledge while mitigating the risk of convergence to sub-optimal policies.
- **Late Phase 2 (85–100%):** The Greedy method becomes dominant, focusing on the stable execution of the learned policy and precise action selection.

### *D. Task assignment strategy*

To enable multiple UAVs to efficiently divide and visit a large number of task points, the proposed method introduces a dynamic task assignment strategy based on learning experience during reinforcement learning. Task points are placed to fixed cells. The proposed method determines which UAVs visit respective task points after a fixed number of training episodes (e.g., 2000 episodes). After the task points are assigned to UAVs, learning process is further continued to optimize the trajectory of UAVs. This assignment is based on a scoring process that utilizes the history obtained during training, such as how many times each UAV visited each task point and the amount of reward obtained during those visits. The UAV with the highest score is then selected to be responsible for the corresponding task point.

The assignment procedure is composed of the following steps:

1) **Score calculation:** For each task point $p$, a score is calculated for each UAV $u$ to evaluate how effectively it has visited the point. The score is defined by:

$$\text{score}(p,u) = \frac{R(p,u)}{N(p,u)} - 0.1 \times d(p,u), \qquad (4)$$

where $R(p,u)$ is the accumulated reward for UAV $u$ obtained by visiting the task point $p$, $N(p,u)$ is the number of times that UAV $u$ visited the task point $p$, and $d(p,u)$ is the Manhattan distance from the starting point of UAV $u$ to the task point $p$. If the task point has never been visited, the score is set to zero. The weighting coefficient of 0.1 is empirically chosen to balance spatial distance and performance history.

2) **Spatial partitioning:** Basically, the proposed method determines the assignment of UAVs to task points based on the score calculated by (4). However, if a task point is clearly close to the starting position of a particular UAV, that UAV is assigned to the task point, regardless of the score.
3) **Task point assignment:** After repeating $T$ training episodes, the proposed method assigns each task point to the UAV with the highest score, where $T$ is a parameter. Once assignment is complete, each UAV stores its designated task points in a private list. During both training and execution, UAVs target only their assigned task points. A significant penalty is imposed for visiting unassigned task points, ensuring clear separation of responsibilities.

In summary, the proposed method achieves flexible and experience-based task allocation through reinforcement learning, while using spatial partitioning as an auxiliary filter. This hybrid strategy enables both efficient and practical task sharing among multiple UAVs.

**TABLE III:** List of hyperparameters.

| Parameter | Value |
|---|---|
| Grid size | $12 \times 12$ |
| Number $N$ of UAVs | 2 |
| Number of episodes | 5,000 |
| Learning rate $\alpha$ | 0.01 |
| Discount factor $\gamma$ | 0.985 |
| Batch size | 32 |
| Target network update interval | Every 40 episodes |
| Parameter $\varepsilon$ in $\varepsilon$-Greedy | Initial: 1.8 $\rightarrow$ Minimum: 0.1 |
| Parameter $\tau$ in Softmax | Initial: 2.0 $\rightarrow$ Minimum: 0.1 |
| Task point assignment threshold $T$ | 2,000 |

## IV. Performance evaluation

### *A. Hyperparameter settings*

In this section, the effectiveness of the proposed method is evaluated both quantitatively and qualitatively. Table III presents the primary hyperparameters employed by the proposed method. In this paper, we assume that there are two UAVs in an area consisting of a $12 \times 12$ hexagonal grid. In the area, there are 5 task points. The UAVs are trained cooperatively to perform a target-reaching task over the course of 5,000 episodes. To enhance learning efficiency, prioritized experience replay is utilized, and the target network is updated at fixed intervals of every 40 episodes. Furthermore, the action selection mechanism integrates both the $\varepsilon$-greedy strategy and the Softmax strategy discussed in Section III-C to dynamically

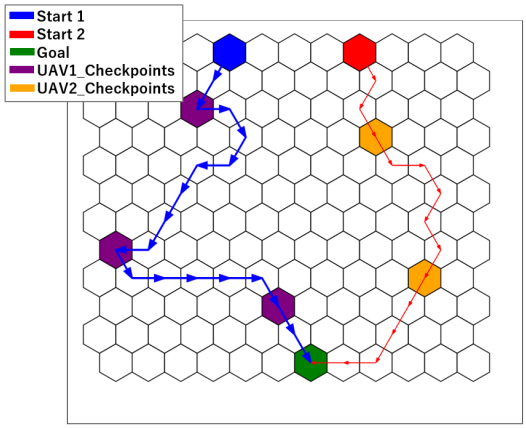


**Fig. 3:** Trajectory of each UAV.

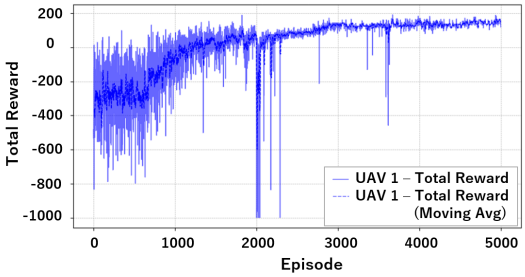


**Fig. 4:** Cumulative reward for UAV1.

manage the trade-off between exploration and exploitation throughout the training process. Note that as shown in Table III, the initial value of $\varepsilon$ exceeds 1, indicating exploration is always performed. As episodes progress, this value decreases, and the exploitation is performed with the probability defined in (2) in the case where this value is less than 1.

### *B. Results*

Fig. 3 illustrates the trajectories taken by each UAV after the completion of training. UAV1 (depicted in blue) and UAV2 (depicted in red) begin from distinct starting positions, visit their respective assigned task points (purple for UAV1 and orange for UAV2), and ultimately arrive at the shared goal location (green). The resulting paths successfully avoid collisions between the UAVs while minimizing unnecessary detours. These results indicate that effective cooperative behavior has been learned. A clear division of task responsibilities is observed, with each UAV autonomously performing its assigned role.

Figs. 4 and 5 show the cumulative episode-wise rewards obtained by UAV1 and UAV2, respectively, throughout the training process. In the early stages of training, the rewards are low and highly variable; however, as learning progresses, they increase and gradually converge to consistently higher levels.

A temporary drop in cumulative rewards is observed immediately after the task assignment update (e.g., around episode 2,000). This decline is followed by a rapid recovery and

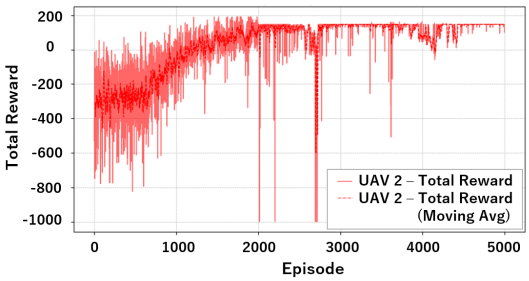


**Fig. 5:** Cumulative reward for UAV2.

subsequent stabilization, indicating that the UAVs successfully adapt to the updated task distribution and continue refining their policies. The transient decrease reflects the system's responsiveness to structural changes in the environment. Furthermore, the strategies, such as replay buffer reinitialization and periodic target network updates, contribute to efficient readaptation and continuous policy improvement.

## V. Conclusion

We proposed a reinforcement learning-based trajectory design method for multiple UAVs for cooperative task allocation. The simulation results demonstrate stable policy convergence, effective task allocation, and efficient trajectory design for the UAVs. Future research will focus on real-world deployment, performance evaluation across diverse mission scenarios, and investigating the scalability of the proposed approach to larger numbers of agents.

## Acknowledgment

This research was partially supported by JSPS KAKENHI Grant No. 23K11077 and 25K15103.

## References

[1] Z. Zhang, Y. Wang, and X. Liu, "Heuristic Task Scheduling on Heterogeneous UAVs: A Combinatorial Optimization Approach," *Journal of Systems Architecture*, vol. 140, 2023. doi:10.1016/j.sysarc.2023.102911.

[2] A. Khochare, Y. Simmhan, F. B. Sorbelli, and S. K. Das, "Heuristic Algorithms for Co-scheduling of Edge Analytics and Routes for UAV Fleet Missions," in *Proc. IEEE INFOCOM*, May 2021, pp. 1–10. doi:10.1109/INFOCOM42981.2021.9488740.

[3] S. S. Khodaparast, X. Lu, and P. Wang, "Deep Reinforcement Learning Based Energy Efficient Multi-UAV Data Collection for IoT Networks," *IEEE Open J. Veh. Technol.*, vol. 2, pp. 45–58, 2021. doi:10.1109/OJVT.2021.3085421.

[4] Y. Wang, Z. Gao, J. Zhang, X. Cao, D. Zheng, Y. Gao, D. W. K. Ng, and M. Di Renzo, "Trajectory Design for UAV-Based Internet-of-Things Data Collection: A Deep Reinforcement Learning Approach," *IEEE Trans. Veh. Technol.*, vol. 71, no. 3, pp. 2345–2358, Mar. 2022. doi:10.1109/TVT.2022.3148364.

[5] Y. Jiao, X. Zhang, W. Liu, Y. Wu, J. Ren, Y. Shen, B. Yang, and X. Guan, "UAV-Enabled Data Collection for IoT Networks via Rainbow Learning," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 4567–4578, May 2024. doi:10.1109/JIOT.2024.3148364.

[6] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "Multi-UAV Path Planning for Wireless Data Harvesting with Deep Reinforcement Learning," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 1171–1187, May 2021. doi:10.1109/OJCOMS.2021.3081996.

[7] K. K. Nguyen, T. Q. Duong, T. Do-Duy, H. Claussen, and L. Hanzo, "3D UAV Trajectory and Data Collection Optimisation via Deep Reinforcement Learning," *IEEE Trans. Commun.*, vol. 70, no. 4, pp. 2358–2371, Apr. 2022. doi:10.1109/TCOMM.2022.3148364.