Construction of 3D Observation Drone Network Using Reinforcement Learning

Ryota Maikuma, Mai Bando, Shinji Hokamoto(Kyushu University)

A network consisting of multiple drones is considered to be a more efficient way to explore large areas. In this study, we use reinforcement learning for generating a "flexible" network, because the drone network should flexibly distribute drones in a three-dimensional space according to exploring environment, network purpose, or the number of drones. In order to reduce the time required for training, two learning procedures are explained and applied to the motion of drones in simulations.

強化学習を用いた３次元観測ドローンネットワークの構築

複数のドローンで構成されるネットワークは、広いエリアを探索するためのより効率的な方法であると考えられている。ドローンネットワークは、環境、目的、またはドローンの数に応じて、各ドローンの位置を３次元で調整する必要があるため、本研究では強化学習を利用して「柔軟な」ネットワークを生成する。学習に必要な時間を短縮するために、２つの学習手順を説明し、シミュレーションでドローンの動きに適用する。

1 Introduction

On July 30, 2020, the first mission operating a drone for Mars exploration has been launched by NASA. The primary objective of the exploration mission is to gather information about astrobiology for much wider area. If this mission is successful, the effectiveness of drone exploration is also approved; done can move extremely fast compared with rovers (typically 5 cm/s), and terrain conditions (roughness, looseness) are meaningless for drones.

Furthermore, in future missions, drone network consisting of multiple drones may be considered as a promising exploration system. From drones' three-dimensional mobility, the network enables wider exploration and detail observation from low altitude. Furthermore, simply increasing the number of drones is effective to achieve both enlarging observation area and repeated observation of interesting points. In addition, multiple drones can in-

crease redundancy or flexibility for the exploration network in case of a drone's machine trouble or leaving for battery charge. Thus, efficient control strategy is necessary for a drone network.

In previous studies on the drone swarm systems for ground observations, the drones are supposed to be in a certain altitude[1, 2, 3]. However, as denoted before, drones can move in any direction and also hover. The mobility is quite important for observation or surveillant task. There are also few studies dealing with the change of the drone number in a network due to malfunctions or battery charges.

The purpose of this study is to construct a more efficient and flexible observation network by considering drones' altitude transitions and the number change of drones. In this study, we aim to allocate drones so that the observation area of each drone does not overlap and the unobserved area is smaller as much as possible.

Applying reinforcement learning on the problems described above in a normal way, the computation time for training would be huge because the numbers of drones and segmented positions are both large. Thus, two learning procedures are explained and applied to the motion of drones in simulations.

2 Q-Learning

In planetary explorations, global positioning systems (such as GPS) cannot be utilized. In addition, computers and other equipments that can be loaded on a drone are limited in terms of weight and size, and their computing ability is also limited. Therefore, in this research, we use reinforcement learning as an approach, which can reduce the online calculation load by learning in advance.

Reinforcement learning is a framework for learning better behaviors of the agents (in this study, drones) from the interactions with the environments.[4] Consider a certain discrete time step t. The drone performs an action a_t at a state s_t, which results in a new state s_{t+1} and a reward r_{t+1} returned from the environment. Based on the above process, selection probability of the action a_t is reinforced so that the total reward will increase.

Q-learning is a typical one of reinforcement learning, and the formula for updating the action value function in Q-learning is as follows:

$$Q(s_t, a_t) \leftarrow$$
$$Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

where α is the learning rate (step size parameter) and γ is the discount rate (weight to future rewards).

For each state and action pair, a value is assigned. The value of one step ago is replaced by using a specified fixed percentage with the reward obtained and the value of the next state action pair.

In this study, multi-agent reinforcement learning process is applied to a group of drones[5].

3 Time reduction method

If Q-learning is applied in discrete state space, the number of states becomes too large

and the learning becomes very slow due to the following reason. For precise position control of drones, state space should be divided into a large number of small subspaces. For example, if the possible positions of each drone are discretized into $10\times10\times10$ grid for m drones, the total number of combination of drones' positions in that case is $(10 \times 10 \times 10)^m$. The computational complexity dramatically increases according to the numbers of subspaces and drones, and thus their increases make the learning process unrealistic. (this is called "the curse of dimensionality".)

Therefore, consider the following two methods – the coordinate-based method and the relative position-based method – to improve the learning process of a multi-drone network. The methods are applicable to different circumstances and purposes of networks.

For the coordinate-based method, s of $Q(s, a)$ is the three-dimensional coordinates of a drone. On the other hand, the action value function table is separately set for each drone; $Q_1, Q_2, \cdots, Q_m$ (m is the number of drones). This is a method called independent Q-learning[5]. In this way, the state quantity in the previous example can be reduced from $(10 \times 10 \times 10)^m$ to $(10 \times 10 \times 10) \times m$.

In the relative position-based method, two types of spaces are defined: a training space and an action (real) space. The training space is divided into equal-sized subspaces through a standard procedure, but the number of subspaces should be relatively small to reduce training time. On the other hand, the action space is divided into the same number of subspaces with the training space, but the sizes of subspaces are not equal. The sizes are defined according to relative positions with neighbor drones in real space. For example, Fig.1 indicates a schematic diagram showing one or two drones placed in 3×3 subspaces in two dimensions for simplicity. In the figure, a drone learning its motion is supposed to be in the center grid, but it is not depicted for better visibility. In the training space, each drone learns better motion according to its position relative to the learning drone. On the other hand, a real space is virtually divided for each drone according to the rela-

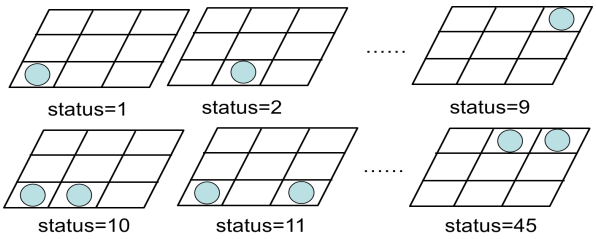

Figure 1: Representation of the drones position in a training space (without learning drone)

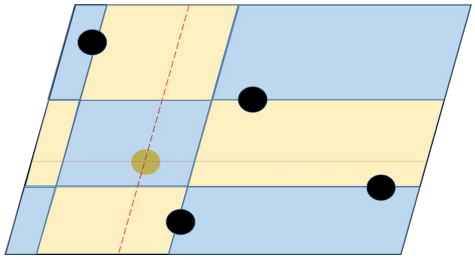

Figure 2: 3×3 grid divisions for the yellow drone in an action space

tive positions with neighbors. For example, Fig.2 shows the subspaces divided into 3×3 in two dimensions for the yellow drone. The action space is discretized according to the relative positions of vicinity drones into the same number of subspaces with the training space. Then, the motion of each drone is decided according to the training results. Although Figs. 1 and 2 are illustrations in two dimensions, the training and the action space divisions are conducted in three dimensions. Note, since the training space is divided into relatively a small number of subspaces, the most time-consuming training process can be shortened.

4 Simulation

First, the simulation conditions of the coordinate-based learning system is described as follows. The number of drones is selected as 3, 5, and 10 drones. The learning rate α is set to 0.04 and the discount rate γ is set to 0.9. The possible states of each drone are set as $10 \times 10 \times 10$, and the possible actions are seven; up, down, left, right, front, back, and wait. As an action strategy, the epsilon greedy action selection[4] is used and the value of $\epsilon = 0.1$. This action strategy is continued even after the learning process is matured to avoid local optimal solutions.

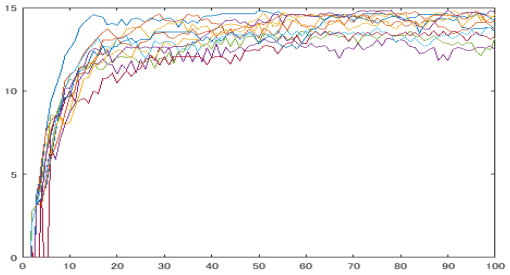

Figure 3: Simulation result with 3 drones (coordinate-based method)

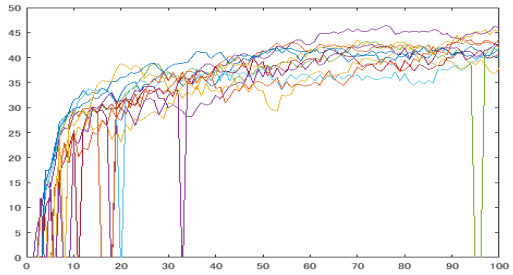

Figure 4: Simulation result with 5 drones (coordinate-based method)

On the other hand, the reward of network systems should be decided according to the desired networks, which can be specified according to the purpose of network and the environment. Thus, in this paper, we simply consider a drone network to be apart from each other as farther as possible. That is, the reward increases as the distance between the drones increases.

The results of the coordinate-based simulation and the reward transition are shown in Fig.3-5 according to drone numbers. Each figure shows results of executed 10 episode. The horizontal axis is the number of iterations, and the vertical axis is the reward calculated from the drone position.

$$\text{reward} = \log\left(\prod_{i \le m, j < i} r_{ij} + 0.01\right)$$

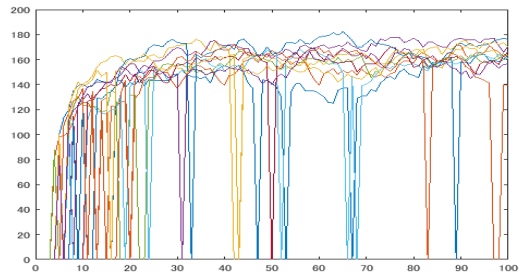

Figure 5: Simulation result with 10 drones (coordinate-based method)

where m is the number of drones, and r_{ij} is a distance between drone i and drone j when each axial length of real space is set to 10. The increase in rewards indicates that learning is being performed properly. In points of the sharp decreases in rewards, the number of drones is relatively large compared to the space where the drones can act, so the distance between the drones is very close.

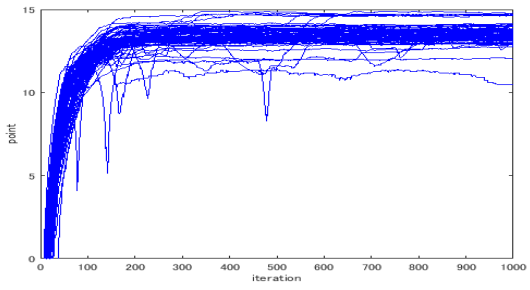

Figure 6: Simulation result (relative position-based method)

Next, the relative position-based learning method is applied to the same problem. The number of drones is set as three, and the training space and also action space are divided into $3 \times 3 \times 3$ subspaces. Figure 6 shows the simulation results of the relative position-based learning system; the horizontal axis is the number of iterations, and the vertical axis is the reward calculated from the drone position in the real space. In this figure, the results of executed 100 episodes are drawn. In the simulation each drone moves along the trained direction but the moving distance is set as 0.1 in one iteration to avoid rapid motion due to the small number of virtually divided subspaces. This figure indicates, although many iteration numbers are required, the final rewards are almost same as Fig.3. It should be noted again that the training cost can be reduced due to a less number of divisions in the training space.

5 Conclusion

This paper has discussed about efficient and flexible observation network by drones. Drones can change the altitude and hover, and the number of drones may be changed in observation missions. Thus, Q-learning is one of promising techniques to generate and keep a flexible network, but reducing the training time for complicated network is requested. Thus, two methods (i.e., coordinate-based method and relative position based method) are explained and their performances are examined in simulations.

References

[1] 長尾 博樹. 神崎 映光. 広範囲の効率的な観測のための自律移動型 UAV の協調動作制御に関する一考察, 第 80 回全国大会講演論文集, (2018) pp.393-394

[2] Julian, K. D. and Kochenderfer, M. J. Distributed Wildfire Surveillance with Autonomous Aircraft Using Deep Reinforcement Learning, JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS, Vol.42 (2019) pp.1768-1778

[3] Yanmaz, E., Costanzo, C., Bettstetter, C. and Elmenreich, W. A Discrete Stochastic Process for Coverage Analysis of Autonomous UAV Networks, 2010 IEEE Globecom Workshops, (2010) pp.1777-1782

[4] Sutton, R. S. and Barto, A. G. Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA (1998)

[5] Tan, M. Multiagent Reinforcement Learning: Independent vs. Cooperative Agents, Proceedings of the 10th International Conference on Machine Learning, (1993) pp.330-337