Path Optimization of a Single Surveillance Drone Based on Reinforcement Learning

Dongoo Lee
Korea Aerospace Research Institute, Daejeon, Korea
Email: ldg810@kari.re.kr

Dowan Cha
Department of Drone-Robot Engineering, Pai Chai University, Daejeon, Korea
Email: chadowan@pcu.ac.kr

***Abstract*—A surveillance drone supervises designated area or sites against dangerous situation. In recent years, it is required to perform autonomous flight to achieve the supervision with path optimization based on minimization of the time lag. In this paper, we propose the reinforcement learning algorithm to optimize path for autonomous flight of surveillance drones. We present a simulation result of a single surveillance drone, which has reinforcement learning algorithm in an unknown grid area. A single surveillance drone finds the optimized path autonomously with minimization of the time lag. This paper provides the following two main contributions for autonomous flight of the surveillance drone. First, the surveillance drone finds the optimized path autonomously using proposed the reinforcement learning algorithm. Second, the traditional reinforcement learning was improved with parameter optimization including learning rate coefficient, convergence criteria, and adaptive error convergence detection for ε-greedy policy process.**

***Index Terms*—path optimization, drone, machine learning, reinforcement learning**

I. Introduction

In recent years, drones are quite active research areas for a civilian or military mission, such as surveillance, delivery, rescue, entertainment, and so on. Kahn et al. (2017), Xie et al. (2017), Perez-Ortiz et al. (2016), Kan et al. (2013), Yang et al. (2015), and Guo et al. (2014) have been studying on autonomous flight, obstacle avoidance or object recognition for mission completion of those drones. [1-6] In particular, it is required for surveillance drones to perform autonomous flight to achieve the supervision with minimization of the time lag and optimized pathing. The time lag means how long the target area remains unsupervised. Surveillance drones supervise designated area or site against dangerous situation. In previous study, Pan (2016), Russell et al. (2003), and Lake et al. developed different approaches including a heuristic approach, a negotiation approach, and an approach based on graph-theory for autonomous flight of surveillance drones. [7-8] However, there were some challenges in terms of control. It was required to equip with many sensors, and quite difficult system dynamics. As a result, a reinforcement learning has become an attractive approach for autonomous flight of surveillance drones recently. The reinforcement learning enables surveillance drones learn the right action without a priori knowledge of environment or its dynamics for autonomous flight. Surveillance drones are able to adapt to changing environment autonomously. In previous study, Bou-Ammar et al. (2010) proposed the reinforcement learning algorithm to control hover and altitude of drones, and Santos et al. (2012) tried to track a defined trajectory using the reinforcement learning algorithm. Lee and Bang. (2011) proposed a model free discrete linear quadratic control tied with Q-learning algorithm, which trained the feedback gain with small errors. [9-11]

In this paper, we propose the reinforcement learning algorithm to optimize path for autonomous flight of surveillance drones. We present a simulation result of a single surveillance drone, which has reinforcement learning algorithm in an unknown grid area. The single surveillance drone finds the optimized path autonomously with minimization of the time lag. We will study issues of surveillance drone in real world, such as the drone dynamics, which is the main issue of a drone while in operation in the future research.

This paper provides the following two main contributions for autonomous flight of surveillance drones. First, the surveillance drone finds the optimized path autonomously using proposed the reinforcement learning algorithm. Second, the traditional reinforcement learning was improved with parameter optimization including learning rate coefficient, convergence criteria, and adaptive error convergence detection for ε-greedy policy process.

The paper is divided into three parts. In Section 2 and 3, we introduce about surveillance mission and surveillance mission based on reinforcement learning, then we provide details of the simulation and the result of

Manuscript received May 1, 2020; revised November 2, 2020.

doi: 10.18178/ijmerr.9.12.1541-1547

the simulation in Section 4. In section 5, we discuss the result and future works.

II. Surveillance Mission

Surveillance drones supervise designated area or site against dangerous situation or enemies. In general, they have been to designated area or site, and they know about terrain information. However, if they have never been to a designated area or site, it is not easy for the surveillance drones to supervise on unknown area or site. In particular, surveillance drones in military or rescue have some challenges to complete their mission successfully on unknown area. In this paper, we represented an unknown area in a grid area as a graph, where the nodes correspond to the specific area or site to supervise. The single surveillance drone should find the optimized path with minimization of the time lag in any unknown environment autonomously. It means a single surveillance drone is able to supervise the designated area against dangerous situation or enemies with autonomous flight. This surveillance mission can be easily mapped to many different domains from computer networks to path planning.

III. Surveillance Mission Based on Reinforcement Learning

The presented surveillance mission was conducted by a drone based on reinforcement learning algorithm. The reinforcement learning is a kind of machine learning that an agent improves its action-policy by exploring the state-space. The details about reinforcement learning will be presented in the next subsection.

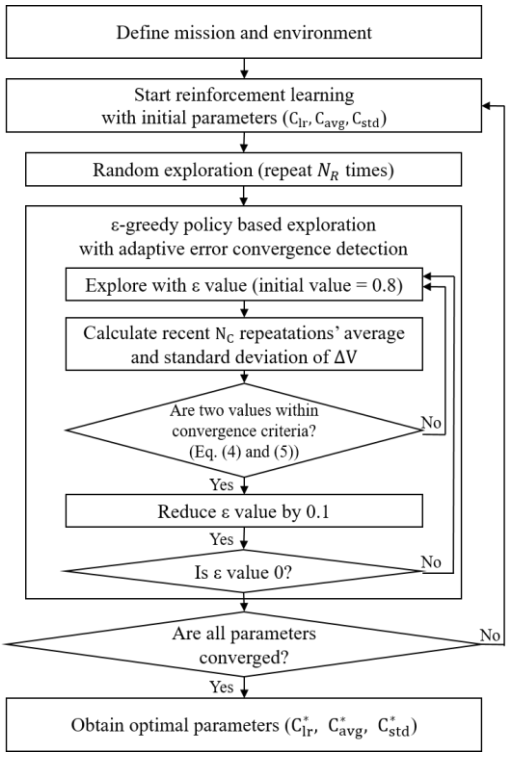

Figure 1. Overall procedure for adaptive error convergence detection-based reinforcement learning

We improve the traditional reinforcement learning approach with parameter optimization and adaptive error convergence detection for ε-greedy policy processes. In the parameter optimization phase, three parameters which are learning rate coefficient (C_{lr}) and two convergence criteria (C_{avg}, C_{std}) are selected as decision variables to be optimized. The learning rate coefficient influences on learning rate α which determines how much newly acquired information would be reflected in agent's behaviour. The two convergence criteria are related with error convergence detection process, which determine whether a learning phase with fixed ε value is converged to a certain state or not. The overall procedures are described in the Fig. 1 and the details about each procedure is provided in the next subsections.

A. Reinforcement Learning

Reinforcement learning which is well summarized by Sutton and Barto (1998) is a kind of machine learning that makes agents in an environment find good actions by collecting more reward with exploration. [12] The reinforcement learning is useful learning technique when there are no correct answers, but agents can calculate reward by taking actions.

In this paper, we modelled an agent in grid area with Markov decision process. The agent with a Markov property solely makes its decision with current state, s, and the value function, $V^{\pi}(s)$ without knowledge of prior state or actions where π is a policy. The agent in this paper will have number of states as $N_P * 2^{N_T} * N_{TS}$ where N_P, N_T, and N_{TS} are number of possible positions, targets and timestamp, respectively.

The value function is updated by the Bellman Equation (1). The E_{π} represents the expected reward value with policy π, γ is a discount factor and r is the reward function.

$$V^{\pi}(s) = E_{\pi}\{\sum_{k=0}^{\infty}\gamma^{k}\cdot r_{t+k+1}|s_t = s\} \quad (1)$$

We update the value function with rule described in (2). The $\alpha(s)$ represents learning rate dependent on state s. The learning rate was set as exponentially decay function as (3) where C_{lr} is learning rate coefficient and N_s is number of times that the state been visited.

$$V(s) \leftarrow V(s) + \alpha(s)[r + \gamma \cdot V(s') - V(s)] \quad (2)$$

$$\alpha(s) = {C_{lr}}^{N_s}, (0 < C_{lr} < 1) \quad (3)$$

B. Action policy: ε-Greedy Policy

We selected ε-greedy policy as the learning policy. The ε-greedy takes random action with ε-percent probability and greedy action which leads to the highest valued state with (1- ε) percent probability. This policy is known as the simplest approach, but very effective learning policy when exploring unknown environment. As learning iteration repeated with fixed ε value, the value function

would be converged to optimal values. Then, we increase the ε value and reset learning rate so that the learning will progress again.

The selection problem of ε is quite old question. A low ε value leads to explore environment randomly and a high ε value leads to exploit previously learned environment data. We set the ε value as 0 in the very beginning of learning phase and increase the value by 0.1 steps until 1.0 whenever the value function becomes optimal state with given ε value. The decision whether the value function reach the optimal state or not will be done with adaptive error convergence detection technique which will be introduced in subsection 3.4.

C. Adaptive Error Convergence Detection for ε-Greedy Policy

To utilize computing resource and terminate learning process when enough learning is already done, we should continuously watch behavior of an agent and detect the error convergence. The technique explained in this subsection which is adaptively detecting error convergence and terminating learning phase is a key contribution of this paper.

The agent would have 10 steps with previously explained ε-greedy policy where the value of ε decrease from 1.0 to 0.0. In this algorithm, the key factor for fast-learning is that how long will each step (phase) be executed. Too fast terminating a phase would result in inadequate learning and too long phase may lead to over-fitted agent with wasting time and computation efforts.

We introduce a technique to detect whether the learning phase should continue or not based on value function difference. The value function, which is updated on every N_u steps where N_u represents a batch size for updating value function, will converge to optimal state as learning progressed. The value function difference (ΔV) means the summed difference of all values in the value function.

The two key factors, average and standard deviation of recent value function difference are used to determine whether a learning phase is converged or not. The range of value function difference for calculating two factors is named as batch size for determining convergence. The size should be large enough to reflect latest behavior of value function differences. Following two equations, (4) and (5) are convergence criteria for i-th learning phase and j-th value function difference value, $\Delta V_{i,j}$.

$$\frac{\sum_{k=j-N_c+1}^{j} \Delta V_{i,k}}{N_c \cdot \max_j \Delta V_{i,j}} * 100 \leq C_{\text{avg}} \quad (4)$$

$$\frac{\sqrt{\sum_{k=j-N_c+1}^{j} (\Delta V_{i,k} - \mu)/(N_c - 1)}}{\max_j \Delta V_{i,j}} * 100 \leq C_{\text{std}} \quad (5)$$

where μ is the average of value function differences from index $j - N_c + 1$ to j.

The general behavior of the value function difference in a learning phase is exponentially decrease with oscillating noise. The reason is that the learning rate is large in the initial section of the learning phases and it becomes smaller as many states are visited. Thus, we selected two factors, average and standard deviation as these are representing the magnitude and noise of ΔV.

D. Parameter Optimization

The learning behavior of an agent will be determined by three key parameters learning rate coefficient (C_{lr}), convergence criteria for average (C_{avg}) and convergence criteria for standard deviation (C_{std}) in the previously explained environment. Therefore, the selection of the three parameters is important to draw a good learning performance of an agent.

We used Nelder-Mead algorithm which was proposed by Nelder and Mead (1965) for optimize parameters. The method is the one of the best-known algorithms to find minimum or maximum of an objective function in multidimensional space. [13] The method makes a simplex which is a convex hull of n+1 vertex in n-dimensional space at initial phase then trying to decrease (or increase) the function value at its vertices. Since the method is not requiring derivatives, it is widely used in many fields.

The performance (= objective function value) of parameter set is calculated by applying greedy approach which selects actions with highest value function direction with final learning states.

IV. Simulation

A simulation with the single drone case is conducted and presented in this section to validate the proposed reinforcement learning for a surveillance mission in the simple grid area. The detailed simulation setup, optimized parameters, and history of value function are presented in next subsections.

A. Simulation Setup

The simulation was conducted on an Intel i7 quad-core personal computer with 16GB RAM under the Windows 7 operating system. The 6 by 6 grid area with three surveillance targets is selected as simulation environment. The environment was constructed using OpenAI gym presented by Brockman et al. (2016) with Python language. [14] The key parameters of simulation environment are summarized in Table I. The number of states (N_{State}) is dependent on four parameters, number of possible positions (N_P), number of target (N_T), number of action (N_A) and number of timestamp (N_{TS}). It is calculated by the following (6).

$$N_{state} = N_P * 2\ N_T * N_A * N_{TS} \quad (6)$$

The agent gets rewards at two conditions. First condition is when the agent reaches a target. The reward is fixed in this condition as 5.0, however, there is distance (time) penalty to impose the agent to find shortest path to a target. The distance (time) is defined as number of

actions took by the agent. The penalty is calculated as number of actions divided by 10 until getting the reward. We selected the number 10 as a denominator of distance penalty to tune the ratio between reward and penalty.

Second condition is when the agent reaches a destination point. The second reward value depends on how many targets were visited before arriving the destination. As the agent found more targets, the reward becomes larger and vice versa. The initial values for parameter optimizations are set as 0.90, 5.0 and 5.0 for learning rate coefficient, convergence criteria for average and standard deviation, respectively.

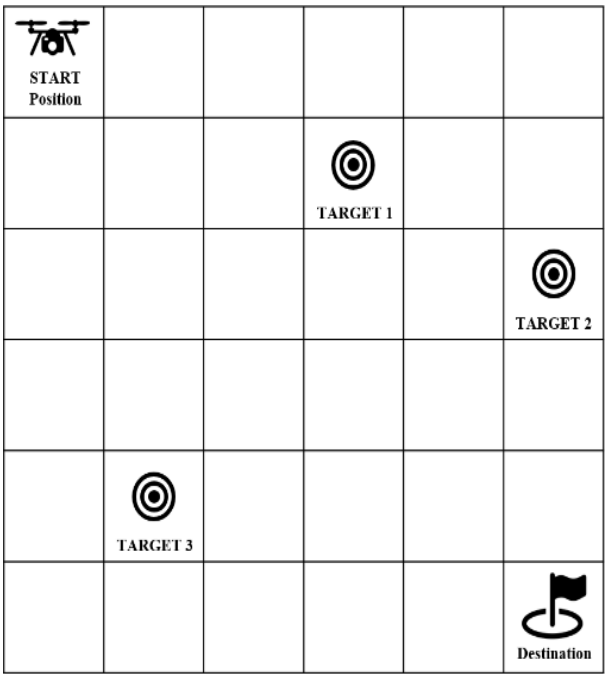

Figure 2. 6 by 6 grid map with start, end position and surveillance targets

B. Simulation Result

The simulation was conducted on an Intel i7 quad core personal computer with 16GB RAM under the Windows 7 operating system. We optimized the learning parameters (C_{lr}, C_{avg} and C_{std}) with tolerance of 0.001 and initial condition in Table I. The final optimized values are presented in Table II. We repeated 500 learnings with the optimized parameters. The average reward was 39.0 and standard deviation of rewards was 2.82. In this environment, the agent can obtain maximum reward of 42.0 (optimal solution). The optimal solution for this case was calculated by brute-force approach as changing the visiting sequence for targets. The number of iterations until final convergence was about 0.25 million and standard deviation was about 21 thousand.

The result of applying simple reinforcement learning method based on the approach of Junell et al. (2015) are presented in the third column of Table II. [15] The learning rate is set as fractional function of number of times that the state has been visited as (7). There are no convergence criteria because it is not adaptively stopping the learning process. The method changes the ε value at fixed number of iterations so it could be called as scheduled ε-greedy method. The average reward of the simple reinforcement learning method was 38.7 and standard deviation was 4.06. The number of iterations was fixed as 0.32 million since it was scheduled learning.

$$\alpha(s) = 1/N_s \tag{7}$$

The proposed method needs about 20% less computational cost than simple reinforcement learning in 500 trials but yielded improved learning result. The average rewards were almost same as 39.0 and 38.7 but the standard deviation of proposed method was much lower than simple reinforcement learning approach which means learning performance of the proposed method is much stable in many tries.

TABLE I. KEY PARAMETERS OF SIMULATION ENVIRONMENT

Property	Value
Number of possible positions, N_p	$6 * 6 = 36$
Number of target, N_T	3 (points where drone should supervise for surveillance)
Number of actions, N_A	8 (left-up, up, right-up, right, right-down, down, left-down, left)
Number of timestamps, N_{TS}	30
Number of random exploration steps, N_R	5,000
Reward at reaching a target	5.0 – penalty (number of action / 10)
Reward at reaching a destination	10.0 * (Number of visited target) – penalty (number of action / 10)
Discount factor, γ	0.99
Batch size for updating value function, N_u	100
Batch size for determining convergence, N_c	2,000 $(= 20 * N_u)$
Initial learning rate coefficient, C_{lr}	0.85
Initial convergence criteria for average, C_{avg} (%)	5.0
Initial convergence criteria for standard deviation, C_{std} (%)	5.0

TABLE II. SIMULATION RESULT

Property	Result of proposed method	Result of traditional method by Junell et al. (2016)
Learning rate coefficient, C_{lr}	0.891	-
Convergence criteria for average, C_{avg} (%)	4.928	-
Convergence criteria for standard deviation, C_{std} (%)	5.051	-
Average reward (500 trials)	39.0	38.7
Standard deviation of rewards (500 trials)	2.82	4.12
Average number of iterations until final convergence (500	250,853	320,000

trials)		
Standard deviation of number of iterations until final convergence (500 trials)	21,384	0

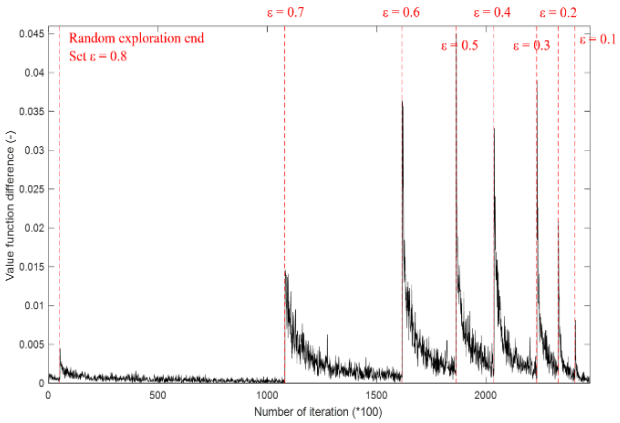

Figure 3. History of value function difference value

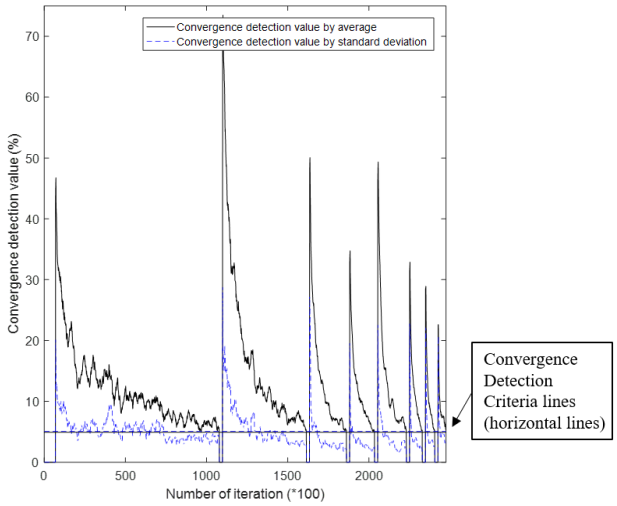

Figure 4. History of convergence detection values (average, standard deviation of recent value function differences)

Fig. 3 and Fig. 4 represents history of value function difference (ΔV) and convergence detection values, respectively. The figures were generated from the same simulation. The value function difference abruptly increases when ε value changed. It means that much information about the given environment is learned in these conditions. As more iteration going, the value decreases and less information is learned. The red dot lines indicate the situations that ε value is changed.

The shape of Fig. 4 is similar with Fig. 3. The values in Fig. 4 presents average and standard deviations of recent value function differences. As the two values become smaller and hit the criteria values (C_{avg} and C_{std}), the current learning phase is converged enough, so new learning phase should be started.

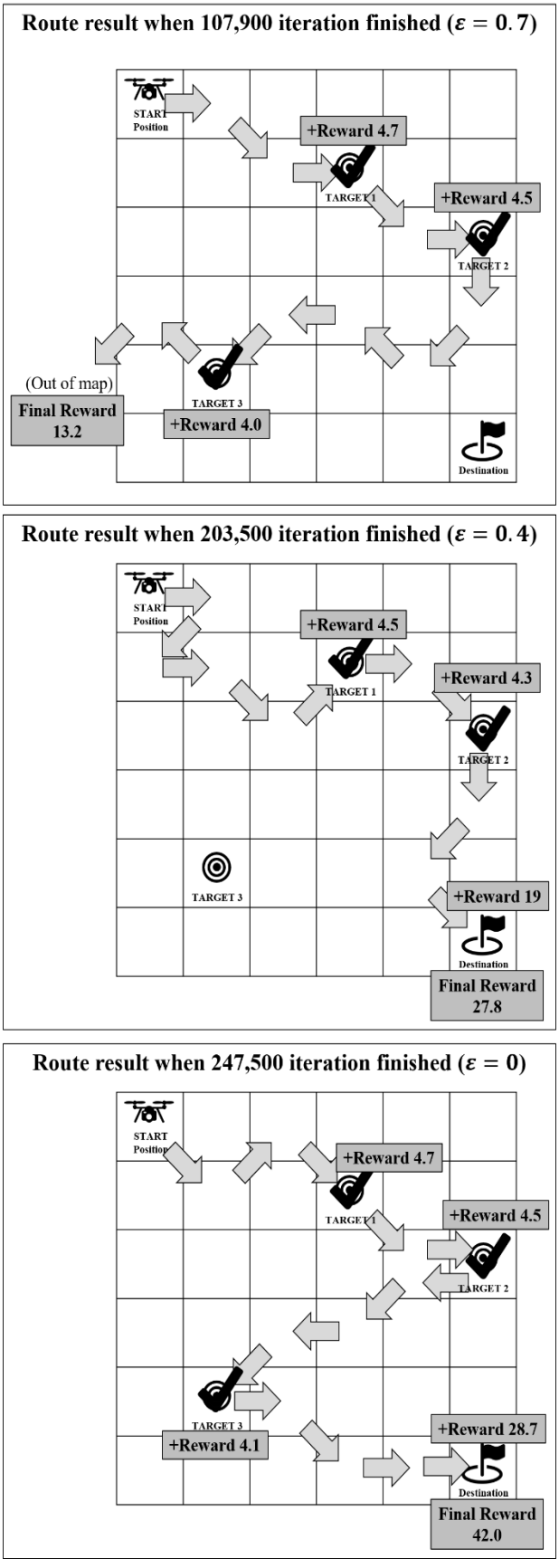

Figure 5. Route history of example simulation

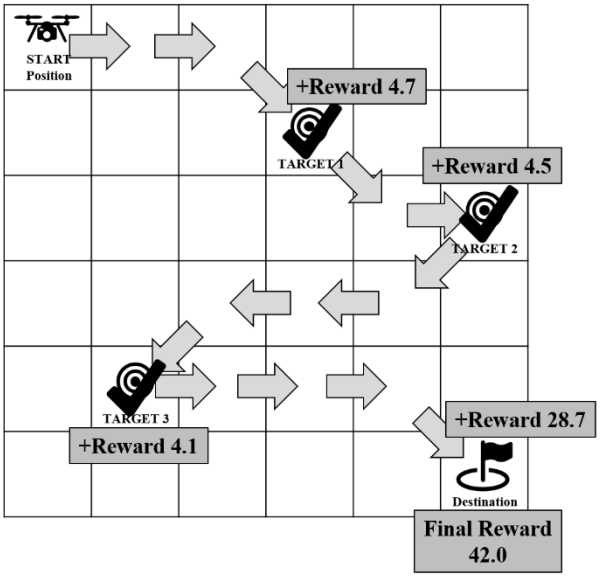

Figure 6. Route and reward history for optimal solution

Fig. 5 below presents the intermediate results of the paths that the agent changes as it proceeds learning phases. In early stage of learning, the agent wanders unnecessarily around target or going out of the map. However, the agent learns new paths to get more rewards at latter stages. Fig. 6 shows optimal solution case when the agent learned perfectly in given environment. The final reward of optimal case is 42.0. The agent could get rewards of 4.7, 4.5 and 4.1 in each target.

V. Conclusion and Future Work

In this paper, we presented a single surveillance drone, which has reinforcement learning algorithm in an unknown grid area. The single surveillance drone finds the optimized path autonomously with minimization of the time lag based on reinforcement learning algorithm. We improved the traditional reinforcement learning approach with parameter optimization including learning rate coefficient and convergence criteria and adaptive error convergence detection for ε-greedy policy process. Simulation results showed that our proposed reinforcement learning approach needs about 20% less computational cost than previous traditional reinforcement learning approach. However, our proposed reinforcement learning approach is required to study more on optimizing learning rate coefficient and convergence criteria for reliability. Also, it is required to study on the single surveillance drone in the unknown real area with real maps to apply it to the real area. As a result, future research will focus on three areas. First, we will optimize learning rate coefficient and convergence criteria in real time. Second, we will research to apply our proposed reinforcement learning algorithm to the unknown real area. Third, we will implement our proposed reinforcement algorithm with a real drone and study the performance of the real drone based on our proposed reinforcement learning algorithm including stability issues while operating.

Nomenclature

C_{lr}	: Learning rate coefficient
C_{avg}	: Convergence criteria for average
C_{std}	: Convergence criteria for standard deviation
N_{State}	: Number of possible states
N_A	: Number of actions
N_c	: Batch size for determining convergence
N_P	: Number of possible positions
N_R	: Number of random exploration steps
N_T	: Number of targets
N_{TS}	: Number of timestamps
N_u	: Batch size for updating value function
γ	: Discount factor
i	: Index for learning phase
j	: Index for value function difference

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Dongoo Lee and Dowan Cha conducted the research; Dongoo Lee and Dowan Cha analyzed the data; Dongoo Lee and Dowan Cha wrote the paper; all authors had approved the final version.

Acknowledgment

This work was supported by the research grant of Pai Chai University in 2020.

References

[1] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, S. Levine, "Uncertainty-aware reinforcement learning for collision avoidance," *arXiv preprint arXiv:1702.01182*, 2017.

[2] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Towards monocular vision based obstacle avoidance through deep reinforcement learning," arXiv preprint arXiv:1706.09829, 2017.

[3] M. Pérez-Ortiz, J. M. Peña, P. A. Gutiérrez, J. Torres-Sánchez, , C. Hervás-Martínez, F. López-Granados, "Selecting patterns and features for between-and within-crop-row weed mapping using UAV-imagery," *Expert Systems with Applications*, vol. 47, pp. 85-94, 2016.

[4] E. M. Kan, M. H. Lim, Y. S. Ong, A. H. Tan, S. P. Yeo, "Extreme learning machine terrain-based navigation for unmanned aerial vehicles," *Neural Computing and Applications*, vol. 22, no. (3-4), pp. 469-477, 2013.

[5] S.Yang, S. Konam, C. Ma, S. Rosenthal, M. Veloso, S. Scherer, "Obstacle avoidance through deep networks based intermediate perception," *arXiv preprint* arXiv:1704.08759, 2017.

[6] X. Guo, S. Denman, C. Fookes, L. Mejias, S. Sridharan, "Automatic uav forced landing site detection using machine learning," in *Proc. 2014 IEEE International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pp. 1-7, 2014.

[7] Y. Pan, "Heading toward artificial intelligence 2.0.," *Engineering*, vol. 2, no. 4, pp. 409-413, 2016.

[8] S. J. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach*, Malaysia; Pearson Education Limited, 2016.

[9] H. Bou-Ammar, H. Voos, & W. Ertel, "Controller design for quadrotor uavs using reinforcement learning," in *Proc. 2010 IEEE International Conference on Control Applications*, pp. 2130-2135, 2010.

[10] M. Santos, J. A. Martin H., V. Lopez, G. Botella, *"Dyna-H : A heuristic planning reinforcement learning algorithm applied to*

role-playing game strategy decision systems," Knowledge-Based System, vol. 32, pp. 28-36, 2012.
[11] D. J. Lee, H. Bang, "Model-free LQ control for unmanned helicopters using reinforcement learning," in *Proc. 2011 IEEE 11th International Conference on Control, Automation and Systems*, pp. 117-120, 2011.
[12] R. S. Sutton, A. G. Barto, *Introduction to Reinforcement Learning*, vol. 135, Cambridge: MIT press, 1998.
[13] J. A. Nelder, R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308-313, 1965.
[14] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," arXiv preprint arXiv:1606.01540, 2016.
[15] J. L. Junell, E. J. Van Kampen, C. C. De Visser, Q. P. Chu, , "Reinforcement learning applied to a quadrotor guidance law in autonomous flight," in *Proc. AIAA Guidance, Navigation, and Control Conference*, p. 1990, 2015.

Dongoo Lee is currently a senior researcher in Korea Aerospace Research Institute. He earned a B.S. in Aerospace Engineering from Korea Advanced Institute of Technology (KAIST; Daejeon, South Korea) in 2012. He received the Ph.D. in the department of Aerospace Engineering from KAIST in 2018. His research interests are design of experiments (DOE) based wind tunnel testing, integrated planetary rover design with routing problems, and artificial intelligence.

Dowan Cha is currently an assistant professor in department of drone-robot engineering at Pai Chai University. He worked at the department of weapon system engineering at Korea Army Academy at Yeongcheon from 2016 to 2019 as the assistant professor. He received the B.A. degree from Korea Military Academy in 2002. He received the MSc. degree in Computer Science from the University of Wales, UK in 2006 and another MSc. Degree in Artificial Intelligence from the University of Wales, UK in 2007. He received the Ph.D. degree in Mechanical Engineering from KAIST in 2014. His research interests include military robots, exoskeletons, the detection of human movement intention, biomechanics, human-robot interface and learning algorithm based on artificial intelligence. He received the Best Session Paper Award from International Conference (ICMERA 2012) held in Romania in 2012, the Best Achievement Award from Brain Korea (BK)21 in 2013, Young Scientist Award from Korea Robot Society(KROS) in URAI 2014, and Outstanding Professor Award (KAAY) in 2018.