Proceedings of the 36th Chinese Control Conference
July 26-28, 2017, Dalian, China

Q learning algorithm based UAV path learning and obstacle avoidence approach

ZHAO Yijing, ZHENG Zheng, ZHANG Xiaoyi, LIU Yang
School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, P. R. China
E-mail: yjzhao@buaa.edu.cn

Abstract: As Unmanned Aerial Vehicle (UAV) having been applied in more complex and adverse environments, the requirements of automatic techniques for obstacle avoidance are becoming more and more important. Reinforcement learning (RL) is a well-known technique in the domain of Machine Learning (ML), which interacts with the environment and learning the knowledge without the requirement of massive priori training samples. Thus it is attractive to implement the idea of RL to support UAV tasks in unknown environments. This paper adopts an Adaptive and Random Exploration approach (ARE) to accomplish both the tasks of UAV navigation and obstacle avoidance. Search mechanisms will be conducted to guide the UAV escape to a proper path. Simulations on different scenarios show that our approach can effectively guide UAVs to reach their targets in quite rational paths.

Key Words: UAV obstacle avoidance, q learning, neural network, trap-escape strategy

1 Introduction

Unmanned Aerial Vehicle (UAV) is widely applied as a kind of multi-functional intelligent carrier today. It can be used in various military missions, such as reconnaissance, attacking and electronic Warfare. Moreover, in civilian aspects, UAV can support geological surveying, mapping, meteorological observation, disaster monitoring, pesticide spraying, etc. For all these tasks, UAV navigation and obstacle avoidance are basic missions. In practical applications, the lack of external environment information and its own accurate state makes these missions difficult to perform. Therefore, researchers have put much efforts on solving environment information to achieve successful navigation.

To date, studies on UAV path planning have been widely conducted with a lot of literatures appearing on modeling and solving methods. The studies related to the achievement of threat information[1] can be divided into two categories: static path planning[2] based on prior complete environment information and real-time path planning[3] studies. Compared with complete environment information, real-time methods assumes that threat environment is partly or completely unknown, which has more practical meanings and attracts more attentions. How to use incomplete information to react to dynamic environment is the key problem in real-time UAV path planning. Many researchers have proposed various artificial intelligent (AI) methods[4], such as Genetic Algorithm[5] and Particle Swarm Algorithm[6]. Through establishing dynamic model[7], reducing computational effort[8] and other ways[9], they have solved some real-time path planning problems with AI methods. However, AI methods can not solve the underlying problem. Because if UAV wants to react to real-time environment accurately, it must develop a dynamic, complex and enormous model which needs long time to obtain a result. Therefore, there is a contradiction between accurate and real-time. Recently, researchers proposed that UAV has become more intelligent by learning which action in its current environment could lead to a better path[10]. To solve the learning problem that UAV reacts properly in a certain environment, RL algorithm[11] has been applied on real-time path planning missions[12] by researchers since 2009. UAV learns how to react by the following mechanism: according to the environment reward, RL algorithm computes the weight of each possible action and obtains the next step. Q learning, a kind of RL algorithm, improved by heuristic searching strategies[13] leads to a smaller action searching space, which reduces searching time and redundant selections. Considering the risk information of other UAVs in the environment, new reinforcement learning methods, such as Cooperative and Geometric Learning[14] and Algorithm Geometric Reinforcement Learning[15] have been introduced into multiple UAVs path planning.These methods have largely improved the adaptability for UAVs in various environments and relaxed many priori limitations.

However, there are still many problems when applying AI approaches to settle path planning tasks. Because it is impossible to obtain all the information in unknown environments, and it is also impossible to predict all the possible things that UAVs will encounter, it is not feasible to use deterministic AI approaches to realize the optimal control of the path planning process. Although some self-learning methods can, to some extent, overcome the problem of lacking prior knowledge, because of various properties of the task scenarios, it probably suffers the problems caused by over-learning. By contrast, less environment knowledge may slow down the learning speed and cause UAV falling into local optimum. Besides, as for the majority of learning approaches, there must be a huge matrix to save the computed value, which may waste considerable storage space.

To solve the above problems, researchers have proposed Artificial Neural Network algorithms(ANN)[16] to solve the problem of large capacity data storage. What's more, ANN always possesses good overall performance on generalization and memory ability[17]. In addition, Monte Carlo Tree method[18], as a stochastic approach, is usually utilized to deal with local optimum, which is supposed to help UAV escape from traps. There is a big successful example, AI robot from Google, Alpha Go[19] use deep Neural Network to make an overall evaluation of the current game state and use the Monte Carlo Tree method to complete the task of rig-

orous calculation. In analogy to Go, for the task of UAV path planning, UAV requires overall heuristics about the current adversarial environment to form rational action directions and, when in extreme situations such as traps, it also need a convergent strategy to form a precise path.

In this paper, we propose the Adaptive and Random Exploration approach (ARE) to deal with the above problem in the task of UAV path planning. The basic idea of ARE is to let the UAV explore the environment itself and make actions according to the current evaluation. In addition, whenever it is close to the obstacles the random mechanism will be conducted to correct it to a safe path. Our approach balance the adaptive mechanism of self-learning and convergent random search so that the subjective UAV can possess both the ability of finding general directions and escaping the dilemma due to learning errors.

The remainder of this paper is organized as follows. Section 2 describes the ARE approaches, including the main process, the associated learning strategies and the semi-random mechanism. Simulations under specially designed scenarios are implemented and analyzed in Section 3. Finally, discussions and conclusions are presented in Section 4 and Section 5.

2 Methodology

2.1 Framework of the ARE Approach

To solve the UAV path planning problem in unknown scenarios, we propose the Adaptive and Random Exploration approach (ARE), which involves both the overall exploration and local calculation for trap-escape strategy. The process of ARE is shown in Fig. 1.

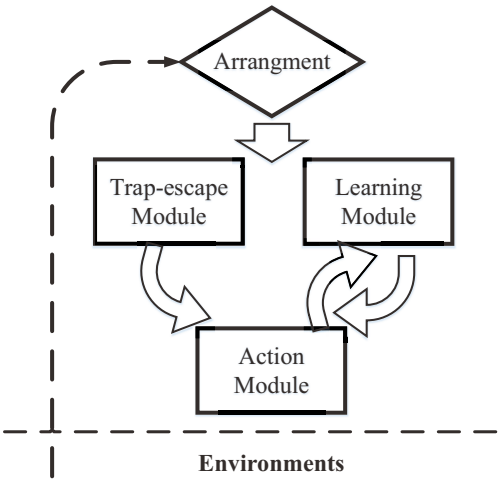


Fig. 1: Framework of the ARE Approach

ARE is composed of three modules, the action module, the learning module, and the trap-escape module. During path planning, the action module receives instructions from the other two modules and let UAV to take actions. The learning module can train the strategy of action selection according to the history data series of UAV states and actions. Finally, the trap-escape module can conduct a random tree search algorithm to guide the UAV escape from the dangerous situation to a safe path. After each step of actions, the state for the UAV in the environment will change. Then an arrangement mechanism will be imposed to evaluate the risks and decide which module should be activated, that is, whether the UAV should escape from the current dilemma or update the action strategy learned from result of the previous action.

Because the UAV do not know the environments at the beginning, we use a q-learning method in the domain of reinforcement learning to do self-learning. And the Local Rapidly Random Tree (LRRT) method will be designed to let the UAV escape from the trap quickly.

In this part, we focus on learning and trap-escape strategies in ARE approach. Learning strategies in the learning module will be introduced in Section 2.2. Then the trap-escape strategy will be introduce in Section 2.3. Finally, we describe the approach overall process by referring a detailed flow diagram.

2.2 Path learning in continuous environmental space

The q-learning methods is conducted to let UAV continuously learn the knowledge about the environment. During path planning, the environmental knowledge is saved by using an updated neural network, which can provide guidance about the next actions. At each step of actions, the action selection mechanism is imposed to select the next action following the guidance.

2.2.1 Q-value function network

Conventional q-learning describes state-action pairs in a discrete space and the $q-$value function can be treated as a map of the index of current state-action pair to the next action index. The state space of path planning is continuous. A direct way is to discretize the continuous state data. However, without the initial knowledge of discretization granularity, the state may have large errors in reflecting the real environments, which may lead to a low quality path and large scale computation of matrix if we want to ensure the precision. Therefore, we use a neural networks to calculated the $q-$value. Instead of the conventional Markov-based evaluation, neural networks save the UAV's current understanding about the environment by several parameters located in different levels, which requires less memory. In addition, according to the mechanism of neural networks, continuous map can be formed considering the connection of different states, which can lead to good learning results as expected.

Define s_t the state of the subject UAV at time t, and a_t the implemented action at time t. Let $x(t) = (s_t, a_t)$ be the state-action pair at time t. Then the history data from time 0 to time t in path planning can be represented by a series of $D^h = (x(1), x(2), ..., x(t))$

Let NN be the neural network adopted in the q-learning strategy. It is trained by the history data and played as the q-value function $Q^{NN}(x)$, i.e.,

$$Q^{NN} : A \to R \tag{1}$$

in which A denotes the set of all the possible actions at the current and R represents the set of preference values for the actions belonging to A. In this paper, we select the BP network with two layers. Q^{NN} is learned based on the history data. During the path planning, the preference values

of each possible actions will be calculated by Q^{NN} at each step. Then the action selection strategy will be conducted according to these preference values.

2.2.2 Action selection strategy

After the evaluation of the preference values of each actions, the UAV should select a final action in the next stage. A direct way is to choose the action with the highest preference value. However, at different stages, due to the complexity of environments and the limitation of learning algorithm, such prediction may not be absolutely accurate. For example, if the scene has a sudden change, the history data will not have much help, or even take negative effect. This may lead to the over training of NN. Consequently, some irrational actions will be assigned with high preference values. In addition, to collect various information of the environment, each action should be given a probability. Therefore, when selecting actions, the higher preference values of the each action, the higher probability it will be selected.

Define A^s the set of actions which are accessible at the current state s, that is, any action $a \in A^s$ is a candidate for selection. Let q_s^a be the preference value of a evaluated by NN at state s. Then , the Boltzmann distribution is adopted in the selection function, which is shown as follows:

$$P(a|s) = \frac{e^{q_s^a/T}}{\sum\limits_{a \in A^s} e^{q_s^a/T}} \tag{2}$$

In the above formula, $P(a|s)$is the probability when UAV is in state s and choose a as the next action. From Eq. (2), if an action a has a high value of q^a, the value of $e^{q_s^a/T}$ will also high, and a will have higher probability to be selected. T is virtual temperature factor, indicating the degree of randomness in the action selection. At the beginning of path planning, the support data is not sufficient for $q-$ learning, and much degree of uncertainty is left; thus T should be set a quite high value to keep data diversity. When the history data becomes sufficient, the value of T should decrease to emphasis the learning result. In addition, when UAV is close to the obstacle, it is reasonable to reduce the T value as well as the length for each step, because current learning result may not be accurate.

2.2.3 Reward and network update

The learning process in path planning is a kind of self-learning. That is to say, the UAV does not know the environment at first, the sample data which support the learning process should be collected as long as path planning. As discussed above, the history data D^h is used as the training samples. For each sample x in D^h, we should assign it a label, indicating whether it is positive or negative. For a state s define d_s^t the distance between s and the final target, and d_s^o the distance between s and nearest obstacle. Suppose $r = (s, a)$ and s' is the state reached by conducting a at s, i.e., the next state of s if a is implemented. In this paper, if r is positive, the value of $d_{s'}^t$ should be lower than d_s^t and $d_{s'}^o$ should be lower than d_s^o. Specifically,the labeling table is shown as follows:

Table 1: Reward of learning action selection

	$d_{s'}^o > d_s^o$	$d_{s'}^o < d_s^o$
$d_{s'}^o < T_{d_o}$	$r = k_1 \times r_1$	$r = k_1 \times r_1'$
$d_{s'}^o \geq T_{d_o}$	$r = k_2 \times r_1$	$r = k_2 \times r_1'$
	$d_{s'}^t > d_s^t$	$d_{s'}^t < d_s^t$
$d_{s'}^t < T_{d_t}$	$r = k_3 \times r_2'$	$r = k_3 \times r_2$
$d_{s'}^t \geq T_{d_t}$	$r = k_4 \times r_2'$	$r = k_4 \times r_2$
$d_{s'}^o < T_{d_{hit}}$	r = - 2	

From Table 1 we divide the in-time situation into four situations, into several categories. For $d_{s'}^o$ and d_s^o, we consider $d_{s'}^o > d_s^o$ and $d_{s'}^o < d_s^o$ to examine whether an action can lead to a safer path. Here, we set a threshold T_{d_o} for $d_{s'}^o$ because it is rational not to consider obstacles if the UAV has enough distance with all the obstacles.For $d_{s'}^t$ and d_s^t, we consider $d_{s'}^t > d_s^t$ and $d_{s'}^t < d_s^t$ to examine whether an action can lead to a safer path. Because if the UAV is close to the target, $d_{s'}^t$ should be paid more attention. Thus we also set threshold T_{d_t} for $d_{s'}^t$.

By classifying the state changes from s to s' as above, we set a reward mechanism: $r = k_i \times r_i$. In this formula, r_i is the unit reward which can be chosen from $\{\mathrm{r}_1, \mathrm{r}_2, \mathrm{r}_{1'}, \mathrm{r}_{2'}\}$. k_i is raising factor depending on the relationship between $d_{s'}$ and T_d. In different conditions, k_i can be chosen from $\{\mathrm{k}_1, \mathrm{k}_2, \mathrm{k}_3, \mathrm{k}_4\}$. Besides, if the value of $d_{s'}^o$ is lower than hitting threshold $T_{d_{hit}}$, UAV can not avoid reaching obstacles by its actions. We set a negative value -2 for r.

After each action a the set of history data will be updated and the NN is re-trained. In this paper, the typical sigmoid function is used to construct neuron cells. And the additional momentum algorithm is adopted for the training process.

2.3 Trap-escape Strategy

In UAV navigation and obstacle avoidance, there are two main problems that can not only be solved by networked q learning algorithm: one is because of action selection probability, UAV may select an action that will hit the obstacles; another is when UAV falls into a local optimum trap, it is impossible to learn by environment reward to explore an trap-escape path.

To deal with the two problems, in this paper we introduce triple trap-escape strategy. First, change the Boltzmann distribution, such as raise temperature parameter T, which can increase the randomness of the action selection policy. Thus it will have a larger possibility to make UAV escape from trap. Second, reduce the step size of algorithm, whose function just like “brake”. Last but not least, we utilize the idea of rapidly random tree(RRT) algorithm to prevent UAV from selecting the action that may cause hitting walls. Therefore, the triple trap-escape strategy can protect UAV from knocking obstacles.

When UAV get the state s, we can know the distance d_s^o between UAV and obstacles. Set up the threshold value T_h larger than UAVs hitting threshold $T_{d_{hit}}$. When d_s^o becomes smaller than T_h, $d_s^o < T_h$, the third trap-escape strategy will play an important role. When $d_s^o > T_h + p$, this strategy stops and quits (p is the step size of UAV). The main policy is following:

We suppose the state s represents UAV position $s(x, y)$ described as one node. UAV starts following trap-escape strategy at initial node s_0. Then it selects arbitrarily a position state s_{rand} who has been produced randomly in the state space. After selection, this strategy explores a node s_1 with step size p along the s_{rand} direction. As a result, we obtain s_1. After repeating the above steps, we can get $s_2, s_3, ..., s_k$ and they form a local random tree. Until the kth tree node satisfies the stop condition: $d^o_{s_k} > T_h + p$, the trap-escape strategy stops and quits. Finally UAV gets a path $s_0 \rightarrow s_k$ which can help UAV escape from the trap and get closer to the target. After this path UAV stays at s_k and continues to explore path by learning network.

2.4 Process of overall ARE Approach

The process of overall ARE Approach can be described as the following steps in Fig. 2:

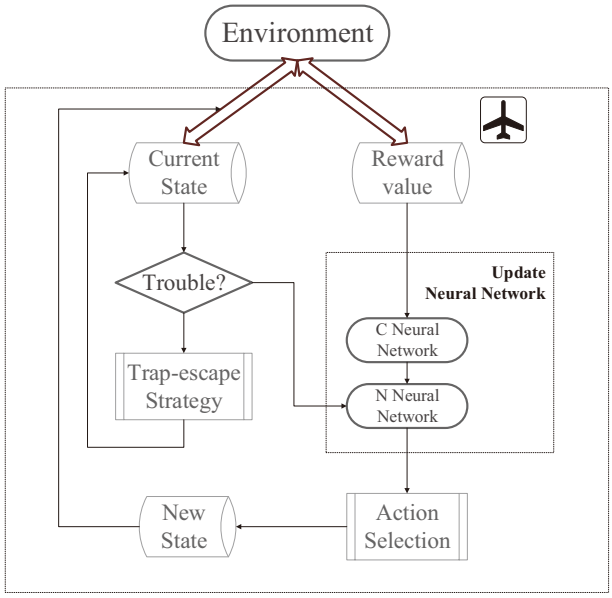


Fig. 2: The method overall flow diagram

Step 1. Initialization. Set the initial state of the UAV, including its position and its knowledge repository.
Step 2. Observe the current state from environment.
Step 3. Judge the UAV state is in trouble or not. If in trouble, use trap-escape strategy to escape from trouble rapidly. If not, go on to action selection.
Step 4. Choose an action, for that state based on one of the action selection policies.
Step 5. Take the action and observe the reward, as well as the new state update the Q-value neural network for the state using the observed reward and the maximum reward possible for the next state.
Step 6. Set the state to the new state, and repeat the process until a terminal state is reached.

3 Simulations and Results

In the simulations, in order to realize the UAV path learning and obstacle avoidance method, we set three algorithm models: q learning model, neural network model and trap-escape model. In addition, we establish the control model and scene model to complete the UAV obstacle avoidance simulations.

3.1 Experiments Setup

Experiments setup includes parameters setup and scene model setup.

3.1.1 Parameters Setup

In the simulation, we assume that sensors can detect objects from the degree 0 to 360 of UAV within 0.5 meter. Nine basic actions are set including turn left/right at the degree 45, 90, 135 and move forward and back which represent the directions of flight.

Neural network used here has one input layer including two state spaces: current state s_t and action index set. Current state contains UAV position, distance from target and obstacles. One hidden layer with 15 neurons is constructed to get an appropriate calculation rate and convergence rate. One output layer with nine neurons is constructed for Q value matrix of nine kinds action selections in the current state. In each learning process, when current state s_t and action index set have been input, we can finally get forecast Q value corresponding to each potential action after calculation by neural network.

We put the Q value matrix into q learning model and make a decision on choosing the next action. After UAV performs an action, environment will feedback a reward to measure whether the last action is suitable or not. On the way to the object, UAV should keep certain distance from obstacles. r_1 and r_2 donate rewards which will be discussed as follows. The total reinforcement value r is the sum of the two parts.

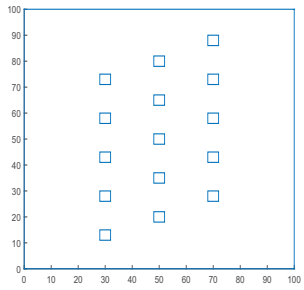
Trap-escape model is different from other models for that it will not be called at each learning process until when it gets into traps. The stop conditions are distributed in two parts: one is $d < F$; the other is distance towards target comes smaller. We identify the stop node as goal point x_{goal}. After getting the trap-escape path, we introduce the path-action function to control the UAV flight following that path. UAV selects a most similar action to nodes from the action set A. Finally we input that action index number $antionIndx$ into control model to finish once function call.

3.1.2 Scene Model Setup

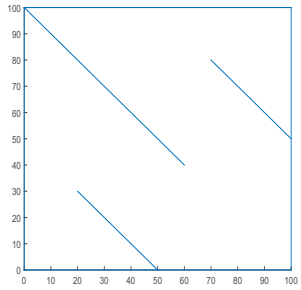
In order to test the effects of learning algorithm as well as trap-escape speed, we establish 4 kinds of obstacle maps. Multi-obstacle array, walls, multi-wall traps and mix obstacles are set respectively in these 4 scenes in Fig. 3. In the first picture, multi-obstacle array is built to test the accuracy of the method. In the second picture, this map can test whether UAV can find a better path. In the third picture, we set some traps hard to escape towards target. In the last picture, we get a mix environment to test the function of method comprehensively.

3.2 Results and Analysis

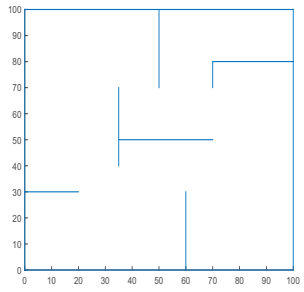
In this part, we introduce the simulated results in different scenes and analyze the progress which is made by this method compared to traditional q learning algorithm.



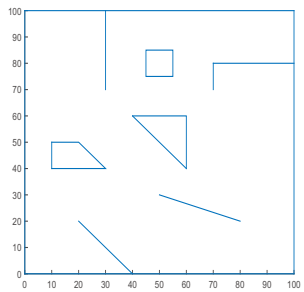
(a) multi-obstacle array



(b) Collateral walls

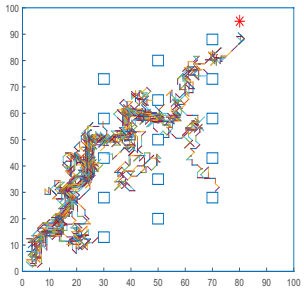


(c) multi-wall traps

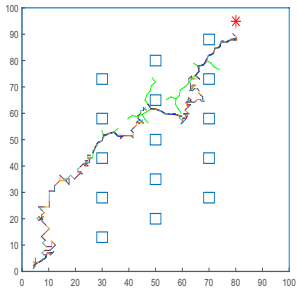


(d) mix obstacles

Fig. 3: Four maps for UAV to test the path learning and obstacle avoidence



(a) path planning based q learning algorithm



(b) path planning based ARE Approach

Fig. 4: Two method comparision in multi-obstacle array map

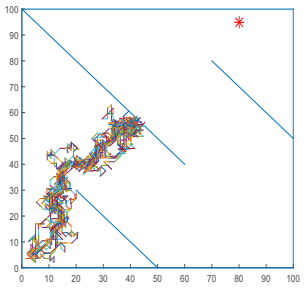
3.2.1 Map One: Multi-obstacle array

Fig. 4(a) shows the result when UAV meet multi-obstacle array based on traditional q learning algorithm. If UAV wants to reach the target it must bypass at least three obstacles. After learning several times, UAV find a path which can avoid all the obstacles towards target finally. But because of its multi-time learning, it is not suitable for online path planning.

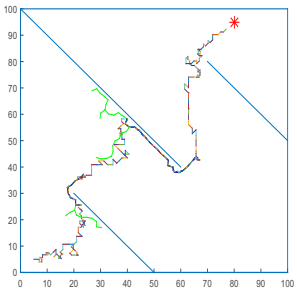
Our method can solve the real-time problem. Fig. 4(b) shows a successful path which can be generated in the first learning process. Therefore, the path learning and obstacle avoidance method can deal with a entirely new map at one time with high efficiency.

3.2.2 Map Two: Collateral Walls

Fig. 5(a) shows the result when UAV meet three collateral walls based on traditional q learning algorithm. After large amounts of attempts and learning, the algorithm is not easy

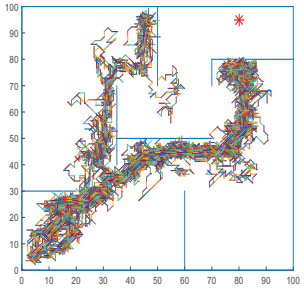


(a) path planning based q learning algorithm

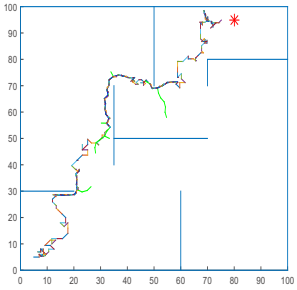


(b) path planning based ARE Approach

Fig. 5: Two method comparision in collateral walls map



(a) path planning based q learning algorithm



(b) path planning based ARE Approach

Fig. 6: Two method comparision in multi-wall traps map

letting the UAV turn around to find a possible way to target only by simple environment rewards.

However, in Fig. 5(b) UAV can learn and plan a complete path without hit walls. When going to meet obstacles, UAV knows how to avoid them and calculate a better path through the method. Especially from this figure UAV bypasses three walls and reaches the target successfully.

3.2.3 Map Three: Multi-wall traps

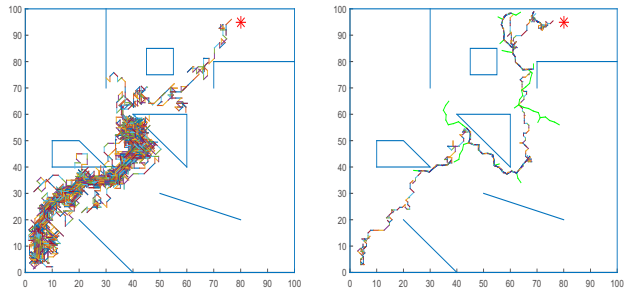
Multi-wall traps map can have more ability to test the UAV's intellectual level. In Fig. 6(a), no matter which directions the UAV tries, after several times learning it will hit walls and fall into local optimal trap. The simple rewards from environment are difficult to solve such problem because that if the UAV desires bypassing the wall-trap, q learning algorithm will tend to prevent it by feed-backing a minus value.

Without setting complex parameters, our method can also obtain a smart path by Trap-escape strategy. Fig. 6(b) shows that based on this method UAV can plan path in an unknown environment at real-time successfully. It makes UAV bypass all obstacle walls and avoid falling into wall traps.

3.2.4 Map Four: Mix obstacles

This map set varies of obstacles including walls, blocks, bricks and traps. From Fig. 7(a) we can finally obtain a practicable path after large amounts of "trial and error". Although we get the successful path towards target, it costs much time and faces large amount of errors.

By utilizing our method, it is obviously fast and accurate



(a) path planning based q learning algorithm (b) path planning based ARE Approach

Fig. 7: Two method comparision in mix obstacles map

to generate an optimal path for UAV real-time navigation in Fig. 7(b). Besides, the method can deal with varies of obstacles for UAV in one map by learning online. So we can utilize this method in unknown environment before we get the knowledge of map.

4 Discussion

This UAV path learning and obstacle avoidance method is the first attempt that achieves on-line path learning while exploring an unknown environment. We establish 4 different maps including walls, blocks, bricks and traps to verify the effects of learning method comprehensively. We can see that we always obtain satisfactory results in learning speed and accuracy.

Although ARE approach was successfully utilized in our research, there are still improvements in some key research directions, such as how to optimize learning parameters in order to obtain better results and how to transfer prior knowledge to similar new learning subjects. Besides, this path learning method can be applied not only to UAV also to driverless car, or any unmanned projects.

5 Conclusion

In this paper the main idea is to solve UAV obstacle avoidance problems. We design a UAV path learning and obstacle avoidance method based on q learning algorithm. On the one hand, Neural network is utilized to achieve continuous state space fitting, which makes UAV easier to learn prior knowledge and improve the learning rate. On the other hand, we suggest a trap-escape strategy to help UAV get out of traps while falling into troubles. In addition, this method have been certificated by simulations in four different maps producing satisfactory results.

From this attempt that utilizes machine learning techniques to solve UAV path planning and obstacle avoidance problems, we find it a promising study direction on this kind of mission. In UAV path planning area we always encounter problems such as uncertain environment. It is too complex and difficult to be considered as a part of objective function in traditional artificial intelligent methods. Therefore, applications on UAV path and mission learning show a promising prospect.

References

[1] L. Zhao, Z. Zheng, W. Liu et al. Real-time path planning for multi-UAVs with share of threats information[C], *Industrial Electronics and Applications (ICIEA)*, 2011 6th IEEE Conference on. IEEE, 2011: 1359-1364.

[2] J. Tisdale, Z. Kim, J. Hedrick. Autonomous UAV path planning and estimation[J]. *IEEE Robotics and Automation Magazine*, 2009, 16(2).

[3] W. Liu, Z. Zheng and K. Cai. Adaptive path planning for unmanned aerial vehicles based on bi-level programming and variable planning time interval. *Chinese Journal of Aeronautics*, 26.3 (2013): 646-660.

[4] E. Masehian, D. Sedighizadeh. Classic and heuristic approaches in robot motion planning-a chronological review[J]. *World Academy of Science, Engineering and Technology*, 2007, 23: 101-106.

[5] Y. Pehlivanoglu. A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV[J]. *Aerospace Science and Technology*, 2012, 16(1): 47-55.

[6] Y. Zhang, L. Wu, S. Wang. UCAV path planning by fitness-scaling adaptive chaotic particle swarm optimization[J]. *Mathematical Problems in Engineering*, 2013, 2013.

[7] U. Zengin, A. Dogan. Probabilistic trajectory planning for UAVs in dynamic environments [A]. In *AIAA 3rd Unmanned Unlimited Technical Conference*, Workshop and Exhibit, Chicago, USA, 2004, pp.1-12.

[8] Z. Peng, B. Li, X. Chen, J. Wu. Online route planning for UAV based on model predictive control and particle swarm optimization algorithm. *IEEE, Intelligent Control and Automation (WCICA)*, 2012, pp.397-401.

[9] K. Klasing, D. Wollherr, M. Buss. Cell-based probabilistic roadmaps (CPRM) for efficient path planning in large environments [A]. In *the 13th International Conference on Robotics*, Jeju, Korea, 2007, pp.1075-1080.

[10] Y. Yang, M. Polycarpou, A. Minai. (2007). Multi-UAV cooperative search using an opportunistic learning method. *Journal of Dynamic Systems, Measurement, and Control*, 129(5), 716-728.

[11] B. Zhang, et al. Cooperative and geometric learning for path planning of UAVs. *Unmanned Aircraft Systems (ICUAS)*, 2013 International Conference on. IEEE, 2013.

[12] W. Liu, Z. Zheng, K. Cai. Bi-level programming based real-time path planning for unmanned aerial vehicles[J]. *Knowledge-Based Systems*, 2013, 44: 34-47.

[13] S. Li, X. Xu, L. Zuo. Dynamic path planning of a mobile robot with improved Q-learning algorithm. *IEEE, Information and Automation*, 2015, pp.409-414.

[14] B. Zhang, W. Liu, Z. Mao, J. Liu, L. Shen. Cooperative and Geometric Learning Algorithm (CGLA) for path planning of UAVs with limited information. *Elsevier, Automatica*, 2014, pp.809-820.

[15] B. Zhang, Z. Mao, W. Liu, J. Liu. Geometric Reinforcement Learning for Path Planning of UAVs. *Springer, Journal of Intelligent & Robotic Systems*, 2015, pp.391-409.

[16] A. Jain, J. Mao, K. Mohiuddin. Artificial neural networks: A tutorial[J]. *IEEE computer*, 1996, 29(3): 31-44.

[17] G. Carpenter, S. Grossberg, J. Reynolds. ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network[J]. *Neural networks*, 1991, 4(5): 565-588.

[18] C. Browne, E. Powley, D. Whitehouse et al. A survey of monte carlo tree search methods[J]. *IEEE Transactions on Computational Intelligence and AI in Games*, 2012, 4(1): 1-43.

[19] D. Silver, A. Huang, C. Maddison. Mastering the game of Go with deep neural networks and tree search[J]. *Nature*, 2016, 529(7587): 484-489.