Real-time obstacle avoidance with deep reinforcement learning * Three-Dimensional Autonomous Obstacle Avoidance for UAV

Songyue Yang
School of Aeronautic Science and Engineering
Beihang University
Beijing, China
yangsy@buaa.edu.cn

Zhijun Meng
School of Aeronautic Science and Engineering
Beihang University
Beijing, China
mengzhijun@buaa.edu.cn

Xuzhi Chen†
†Corresponding author at:
School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China.
Tel:+86 13488680026
chenxuzhi256@126.com

Ronglei Xie
School of Aeronautic Science and Engineering
Beihang University
Beijing, China
xieronglei@buaa.edu.cn

ABSTRACT

At present, drones are rapidly developing in the aviation industry and are applied to all aspects of life. However, letting drones autonomously avoid obstacles is still the focus of research by aviation scholars at this stage. However, the current automation is mostly based on human experience to determine the obstacle avoidance strategy of UAV. And the method only rely on the machine to avoid obstacle is very few. In this paper, the UAV collect visual and distance sensor information to make autonomous obstacle avoidance decision through the deep reinforcement learning algorithm, and the algorithm is tested in the v-rep simulation environment.

CCS CONCEPTS

• Theory of computation • Computing methodologies ~ Motion path planning • Computing methodologies ~ Reconstruction • Computing methodologies ~ Concurrent computing methodologies

KEYWORDS

aircraft, obstacle avoidance, DQN, v-rep

1 Introduction

At present, in order to make the UAV adapt to the unknown environment intelligently and autonomously, it is necessary to make the UAV sense the external environment independently and make corresponding autonomous response to the complex external environment. There are many obstacle avoidance methods, which are mainly divided into two steps. First, the sensor information is used to obtain obstacle information around the obstacles, and then the traditional path planning method is used to avoid the surrounding obstacles. Visual sensation is usually used to recover reconstruction obstacles. Common methods are optical flow[1,2], detection of vanishing points[3], visual SLAM[4] and segmenting traversable areas based on visual appearance[5]. All of the above methods can extract the obstacles from the visual information, so as to navigate the agent avoiding obstacle through the traditional path planning method [6]. However, traditional methods[7-10] require manual adjustment of a large number of parameters. In recent years, reinforcement learning has gained more and more research applications, and various algorithms based on DQN have been gradually developed [11-18]. Reinforcement learning uses the method of acquiring samples while learning, and uses the obtained samples to update its model in order to guide the next action, and iterates until the model converges. Reinforcement learning can be applied to the autonomous flight of the aircraft

In this paper, the modeling and simulation of the common obstacle scenarios of drones are carried out in the v-rep simulation environment. Several algorithms based on DQN are used to perform visual-based obstacle avoidance experiments, and the adaptation and robustness of various algorithms to visual obstacle avoidance is studied.

2 Relate Works

Deep Reinforcement Learning(DRL) can interact with the agent and the environment through trial and error, and can continuously interact according to the set reward value to obtain the maximum cumulative reward, so as to obtain the approximate optimal strategy. Agents usually obtain observed state values S_i from the environment. Depending on state S_i, agent adjusts strategy π to do the right action A_i and feedback to the environment. According to the action A_i ,environment gives a return value R_i to the Agent. Iterate over and over to get appropriate strategies. As shown in the figure:

RICAI '19, September 20–22, 2019, Shanghai, China

ACM ISBN 978-1-4503-7298-5/19/09…$15.00
https://doi.org/10.1145/3366194.3366251

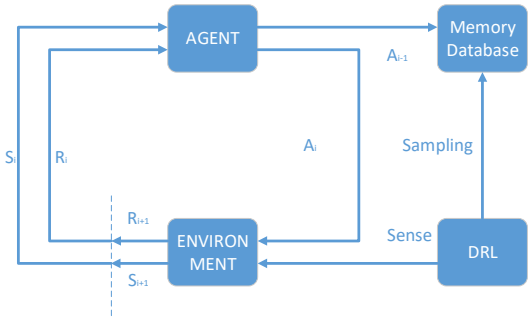

Figure 1: DRL

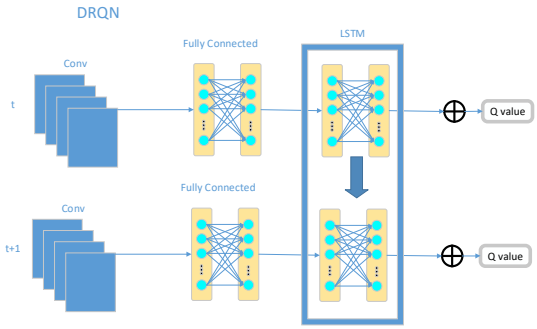

Figure 2: DRQN

Mnih [11] et al. combined convolutional neural network(CNN) with traditional reinforcement learning algorithm Q-learning, proposed a algorithm, deep Q-network(DQN), which solved the problem that reinforcement learning used neural network to approximate value function in large state space and resulted in instability. DQN adopts experience playback mechanism on the basis of traditional Q learning. The sample $\exp_i=(s_i, a_i, r_i, s_{i+1})$ obtained from each online interaction is not immediately trained, but is stored in a memory $M_i = (\exp_1, \exp_2, \dots, \exp_i)$. Each training time randomly takes a certain number of samples from the memory M for training, so as to ensure that the data of the input neural network is independently and there is no correlation between the samples.

Nature DQN fits the current value function and the target value function respectively by setting two neural networks with the same structure but different parameters (θ and θ'). By setting up the target network to deal with the TD-error of the time difference algorithm, the convergence of the algorithm is guaranteed.

$$L = E\left(\left[r + \max_{a'} Q(s', a'|\theta') - Q(s, a|\theta)\right]^2\right) \quad (1)$$

Dueling-DQN separates the display of original single-valued action value function into state value function V(a) and action advantage function A(beta) in the final full connection layer. The combination of the two functions acts as action value function. The Dueling mechanism can obviously accelerate the convergence of deep reinforcement learning under multi-action.

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \alpha) + A(s, a; \theta, \beta) - \frac{1}{|A|}\sum_{a'} A(s, a'; \theta', \beta) \quad (2)$$

Compared with the original DQN, Deep Recurrent Q Network(DRQN) can process the information of time dimension better by introducing the recurrent neural network LSTM, which has better perception ability for time series. Each time step only needs to input a single picture instead of multiple pictures for learning. At the same time, because LSTM layer can memorize historical information, it can improve the performance of the algorithm in some partially observable scenes.

3 Network Model Architecture

Before the formal image processing, it is generally necessary to pre-process the image, extract the main effective information from the complex information to reduce the complexity of the data and improve the efficiency of the algorithm. Firstly, the 256*256 color image in the simulation environment is converted into gray-scale image. In this experiment, using gray-scale image will not affect the performance after learning, and can effectively improve the computing speed. Then, the 256*256 gray-scale image is interpolated bilinear, so that the image is scaled to 80*80 gray-scale image. Finally, the 16*1 distance information and 80*80 image information are contacted into a picture as the state s.

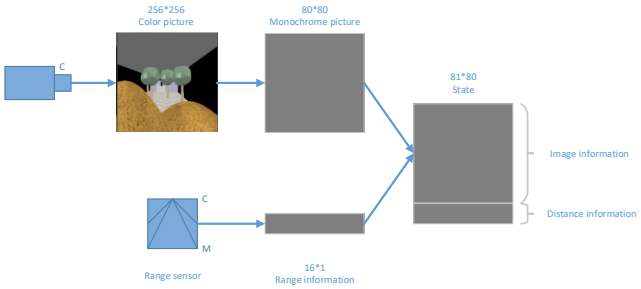

Figure 3: State for RL

Through the fitting ability of the CNN, the contacted information is dimension-degraded, and the information in the image is extracted layer by layer using three-layer convolution and one fully connected network. The parameters are shown in the table below:

Form 1 The layers of network

Layer	Input	Filter	Stride	Activation	Output
Conv1	1*81*80	8*8*32	4	ReLu	20*20*32
Pool1	20*20*32	2*2	1	/	10*10*32
Conv2	10*10*32	4*4*64	2	ReLu	5*5*64
Conv3	5*5*64	3*3*64	1	ReLu	5*5*64
FC1	1600	/	/	/	512

For the reduced-dimension image and distance information already obtained by the neural network, further processing is needed to obtain the action value function of each action. The DQN directly obtain the Q value corresponding to each action by decoding the dimension-degraded information through fully connected layer. And the Dueling-DQN obtains the A and V values respectively through two networks with different structures, and the sum of the two is taken as the Q value. The RL makes a decision through the obtained Q value to decide which action to take to avoid obstacles. At the same time, the network parameters are trained by back-propagation, so that the Q value approaches the value of the reward high.

In this paper, three different algorithms, Nature-DQN, Dueling DQN and DRQN, are used to obtain Q values, and compare the obstacle avoidance capabilities of different algorithms.

4 Experimental setting

This section introduces the hyper-parameters set in the experiment, these parameters determined the performance of the agents. Then, according to the UAV's performance and flight environment, a simulation model platform is set up and established.

4.1 Setting of experimental parameters

In order to effectively compare the advantages and disadvantages of each algorithm, we uses the same set of parameters for different algorithms.

Form 2 The Hyper-parameter

Hyper-parameter Name	Value	Meaning
Minibatch	32	Number of sampled transition from Memory each step
N	300	Update frequency of the target network
Replay_Memory	50000	Number of previous transitions to remember
λ	0.99	Decay rate of past observations
α	3e-6	Learning rate
Observe steps	200	The steps agent fills the memory
Explore steps	200000	The steps ε Change from $\varepsilon_{init}$to ε_{final}
$\varepsilon_{init}/\varepsilon_{final}$	0.3/0.01	The probability agent randomly selected an action
$decay_{num}$	1e-4	The decay of suggest-num every step

4.2 UAV simulation environment settings

For different agents, obstacle scenarios need to be modeled according to their own characteristics. In view of low speed small unmanned aircraft slow moving speed and low flying altitude, the action space of the agent is discretized and the obstacle model which is in accordance with the characteristics of flight environment is established.

4.2.1 Discretization of UAV action space. Horizontal action space discretization: According to the restriction of UAV turning radius, the horizontal flight time step length is determined. Setting the horizontal flight time step length of UAV as Rad_turn, we make UAV fully maneuverable in the minimum state space. As shown in the figure, black grids represent obstacles and white areas represent accessible areas. Vertical action space discrete: According to the UAV climbing ability, set reasonable vertical space action, because the low speed small UAV climbing speed is relatively slow, so only three vertical actions are taken, namely, horizontal, climbing and descending flight.

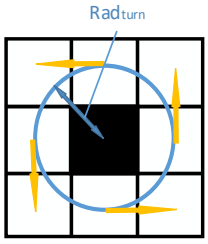

Figure 4: Horizontal spatial dispersion

4.2.2 Discretization of UAV action space. There are many sensors commonly used to avoid obstacles on drones. In this paper, the IN-SIGHT series of visual sensors are used to obtain 256*256 picture information and the laser distance sensor to obtain 16*1 distance information. The combination of the two to obtain information on the front obstacles.

4.2.3 Return function settings. The essence of reinforcement learning is to minimize the loss function in order to obtain a strategy that can obtain a larger reward. According to various performance indicators of the drone, a variety of rewards are set to perform weighted summation to ensure that an approximate optimal track is finally obtained.

Form 3 The kinds of rewards used in this paper

Flight distance reward
$r_{dis} = \sqrt{(x_{tar}-x_{start})^2+(y_{tar}-y_{start})^2+(z_{tar}-z_{start})^2} - \sqrt{(x_{tar}-x_{now})^2+(y_{tar}-y_{now})^2+(z_{tar}-z_{now})^2}$
The drone flight height penalty reward
$r_h = \lvert z_{now}-h\rvert$
Punishment reward of flight fuel consumption
$r_{oil} = w_{hor}^{oil} dis_{hor} + w_{ver}^{oil} dis_{ver}$
Collision and finish reward
$r_{terminal} = \begin{cases} +1.5 & \text{reach target} \\ -1.5 & \text{crash} \end{cases}$

The Form 4 is the reward items for training RL model. Flight distance reward is to approach the target return, the closer to the target point, the higher the return value; The drone flight height penalty reward is to keep the aircraft flying at a certain altitude; Punishment reward of flight is to avoid obstacles for the aircraft with less fuel consumption; and Collision and finish reward is the penalty of collision and reward after reaching the target point,

which is to make the UAV reach the target point without crashing. $(x_{tar}, y_{tar}, z_{tar})$ is the target point coordinate; $(x_{star}, y_{star}, z_{star})$ is the starting point coordinate;$(x_{now}, y_{now}, z_{now})$ is the current aircraft coordinate; $(w_{hor}^{oil}, w_{ver}^{oil})$ is the fuel consumption of horizontal and vertical movement, (dis_{hor}, dis_{ver}) is the distance of horizontal and vertical movement.

The final reward function is as follows, where w_i is the weight value of the different reward functions:

$$r = \begin{cases} r_{terminal} & terminal \\ w_0 * \log(r_{dis}) + w_1 * r_h + w_2 * r_{oil} & else \end{cases} \quad (3)$$

4.2.4 Training obstacle avoidance scene construction. Aiming at the common obstacles of UAV, we build a scene of obstacles suitable for UAV flight environment on the vrep simulation platform, and set up four kinds of common obstacles between the starting point and the end point, such as soil pile, trees, windows and stone piers. Obstacle scenarios are divided into training scenarios and test scenarios. The training scenarios contain all four types of obstacles. The test scenarios change the order of obstacles without changing obstacle categories.

5 Experimental evaluation and result analysis

In the obstacle scene, we use the several algorithms proposed in this paper to test the convergence of the training scene, as shown in the figure.

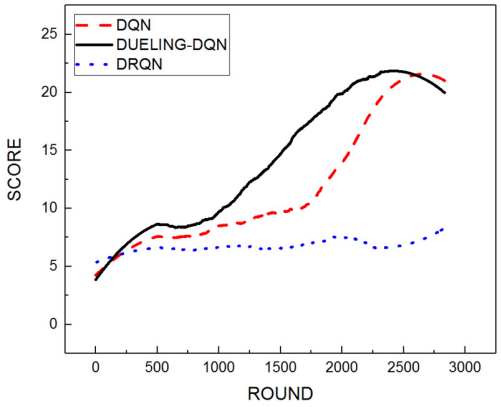

Figure 5: The score for the five model on training scene while training

The convergence speed of DRQN series algorithm is much lower than DQN series. Although DRQN has added LSTM layer, it is sensitive to the time series information obtained by exploring. Compared with multi-frame DQN, DRQN does not produce systematic improvement. Due to the introduction of timing structure, the information of previous time frames needs multiple iterations to be processed before it can be transmitted into the current cell. Although more time frames information can be obtained, it is also difficult to fit obstacle avoidance strategies of various obstacles. Although Hausknecht [13] and others use DRQN to achieve better results in time-based games such as Pong, the use of images for UAV obstacle avoidance is different from the Atari2600 game. Due to based on the first person perspective, the visual input for the obstacle avoidance of drones is more complex. DRQN is difficult to accurately locate the obstacle location in complex scenes, which makes the agent's obstacle avoidance ability worse.

Unlike the two-dimensional obstacle avoidance, the flight action of aircraft is more than the vehicle of two-dimensional obstacle avoidance because of obstacle avoidance in three-dimensional space. Dueling-DQN divides the original action value function into two branches, the action advantage function determines the optimal action, and the state value function determines the whole state Q value. In Dueling-DQN, multiple actions share the same state value V and the action is determined by action advantage A, which makes it possible for the whole model to converge more quickly.

Using the trained weights, the trained weights are tested in the training scenario. All kinds of DQN algorithms can converge and reach the target point. For four kinds of obstacles, DQN series algorithm can fit the appropriate obstacle avoidance strategy. When the front is unobstructed, the Q value changes relatively smoothly.. Once obstacles appear, Q value will increase correspondingly. For different obstacles, Q value will change differently. A encountered obstacles - mounds, it can be seen that the Q value of various algorithms increased slightly, and then decreased after the obstacle; then the tree barrier at B, due to the spacing between the tree and tree, makes the Q value appear a gentle high slope; because the window at point C is the most complicated obstacles, it is the most difficult to obtain a suitable strategy, which makes the Q value variation obvious.

At the same time, carefully observe the obstacle avoidance path, it easy to find that in the final stage of training scenario and test scenario 1 the aircraft obstacle avoidance strategy appeared "errors". When passing through stone piers, the algorithms tend to fly up or down, rather than fly near the optimal flight altitude and choose the optimal path. In test scenario 2, it is found that when the aircraft passes through the first stone piers, it chooses to fly near the optimal flight altitude, but when it passes through the second stone piers near the final target point, a "sub-optimal" obstacle avoidance strategy appears. After analysis, agent mistakes "target point" as "obstacle" near the final target point which makes it adopt the strategy of "avoiding obstacle" and makes the final path worse.

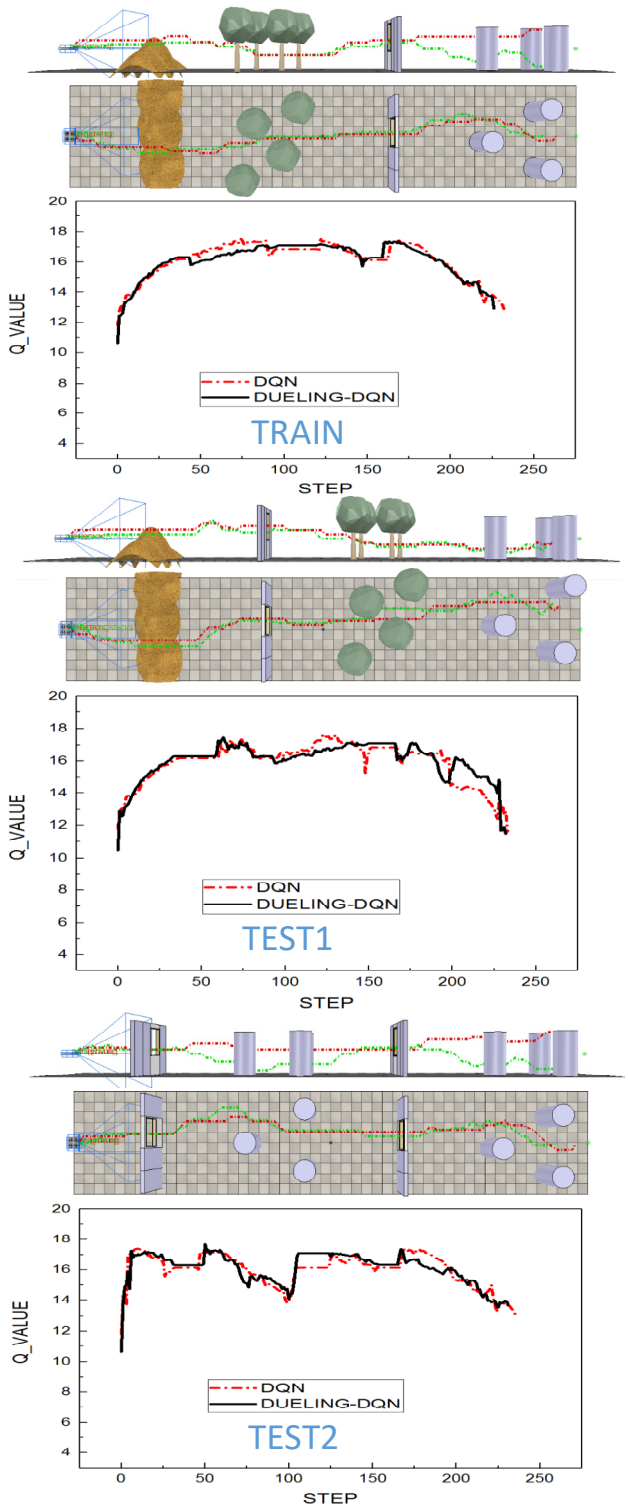

Figure 6: The test on scenes after training

Form 5 Performance comparison of different algorithms

Algorithm		DQN	Dueling-DQN
Color		RED	BLACK
Round average reward	Train	0.1510	0.1483
	Test1	0.1518	0.1472
	Test2	0.1511	0.1413
Average Q_value		15.9958	16.0740

The weights of the original algorithm and the improved algorithm are trained respectively. All kinds of algorithms can reach the final goal through the trained weights in the training and testing scenarios, which proves that the algorithm can be applied to real-time obstacle avoidance. Although Dueling-DQN can ensure that obstacle avoidance strategies converge faster and reach the final target faster under the same training steps, the return value of Dueling-DQN is lower than that of Nature-DQN. It can be seen that the obtained strategy by Dueling-DQN is not as good as the Nature-DQN algorithm obviously.

6 Conclusion

This article has two major contributions. Firstly, the possibility of using deep reinforcement learning in UAV obstacle avoidance is verified, and the real-time ability is proved by testing the non-training scenarios in v-rep. Secondly, two algorithms, Nature-DQN and Dueling-DQN, are tested, and it is found that the strategies learned by Nature-DQN are better, but the convergence rate is not as fast as Dueling-DQN.

ACKNOWLEDGMENTS

This work has taken place in the Beihang University in Beijing, China. The research is supported by the National Natural Science Foundation (NSF) of China (No. 61702023, No. 91538204) and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] Grau, A. . Optical flow based robot obstacle avoidance with matlab. Computer Vision.
[2] Mccarthy, C., & Bames, N. (2004). Performance of optical flow techniques for indoor navigation with a mobile robot. IEEE International Conference on Robotics & Automation.
[3] Bills, C., Chen, J., & Saxena, A. (2011). Autonomous MAV flight in indoor environments using single image perspective cues. IEEE International Conference on Robotics & Automation.
[4] Mur-Artal, R. , Montiel, J. M. M. , & Tardos, J. D. . (2015). Orb-slam: a versatile and accurate monocular slam system. IEEE Transactions on Robotics, 31(5), 1147-1163..
[5] Ulrich, I., & Nourbakhsh, I. R. (2000). Appearance-based obstacle detection with monocular color vision. Aaai.
[6] Abtahi, F., & Fasel, I. (2011). Deep belief nets as function approximators for reinforcement learning.
[7] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(3), 533-536.
[8] Zhou, S., Chen, Q., & Wang, X. (2014). Fuzzy deep belief networks for semi-supervised sentiment classification. Neurocomputing, 131(9), 312-322.
[9] Riedmiller, M. . (2005). Neural fitted q iteration – first experiences with a data efficient neural reinforcement learning method..
[10] Lange, S., & Riedmiller, M. (2010). Deep auto-encoder neural networks in reinforcement learning. International Joint Conference on Neural Networks.
[11] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., & Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529..

[12] Van Hasselt, H., Guez, A., & Silver, D. (2015). Deep reinforcement learning with double q-learning. Computer Science.
[13] Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Freitas, N. (2015). Dueling network architectures for deep reinforcement learning.
[14] Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. Computer Science.
[15] Narasimhan, K. , Kulkarni, T. , & Barzilay, R. . (2015). Language understanding for text-based games using deep reinforcement learning. Computer Science, 40(4), 1-5.
[16] Hausknecht, M., & Stone, P. (2015). Deep recurrent q-learning for partially observable mdps. Computer Science.
[17] Lample, G., & Chaplot, D. S. (2016). Playing fps games with deep reinforcement learning.
[18] Schulze, C., & Schulze, M. (2018). Vizdoom: drqn with prioritized experience replay, double-q learning, & snapshot ensembling.
[19] Tsitsiklis, J. N., & Van Roy, B. (2002). An analysis of temporal-difference learning with function approximation. IEEE Transactions on Automatic Control, 42(5), 674-690.
[20] Hausknecht, M., & Stone, P. (2015). Deep recurrent q-learning for partially observable mdps. Computer Science.