HER-PDQN: A Reinforcement Learning Approach for UAV Navigation with Hybrid Action Spaces and Sparse Rewards

Liu, C.; van Kampen, E.

DOI
10.2514/6.2022-0793

Publication date
2022

Document Version
Final published version

Published in
AIAA SCITECH 2022 Forum

AIAA SciTech Forum
January 3-7, 2022, San Diego, CA & Virtual
AIAA SCITECH 2022 Forum

10.2514/6.2022-0793

HER-PDQN: A Reinforcement Learning Approach for UAV Navigation with Hybrid Action Spaces and Sparse Rewards

Cheng Liu * and Erik-Jan van Kampen †
Delft University of Technology, Delft, the Netherlands, 2629HS

Reinforcement learning (RL) equipped with neural networks has recently led to a wide range of successes in learning policies for unmanned aerial vehicle (UAV) navigation and control problems. The success of RL relies on two human-designed heuristics: appropriate action space definition and reward function engineering. The commonly used fully continuous or fully discrete action spaces in optimal control and decision making problems may lack control authority and remove the inherent problem structure, which can negatively affect learning performance. Besides, reward engineering requires a lot of human effort and may lead to unwanted behavior. In this paper, we address these challenges by proposing a new off-policy RL algorithm called HER-PDQN which incorporates Hindsight Experience Replay (HER) with Parameterized Deep Q-Networks (P-DQN). In simulation experiments, HER-PDQN is used to train an agent to fulfill a UAV navigation task in a 2-dimensional environment. The results indicate the effectiveness of P-DQN algorithm in dealing both with the hybrid action space and sparse rewards. This paper can be considered as the first attempt at applying RL in sparse reward setting for UAV navigation with hybrid action spaces.

I. Introduction

Over the past few years, we have seen an unprecedented growth of unmanned aerial vehicles (UAVs) applied to various areas [1–3]. A long term goal of UAV applications is to build intelligent systems that can implement various tasks such as guidance and navigation without human intervention. A promising method to approaching this goal would be reinforcement learning (RL) [4], a field of machine learning where agents learn a policy to decide what actions to take by interacting with the environment. RL enables us to tackle complex decision making and optimal control problems both in robotics [5–8] and UAV areas [9–12]. When defining the corresponding RL framework in these problems, the action space is commonly approximated with fully continuous or fully discrete ones. However, there exists situations that the RL agent needs to deal with both discrete and continuous action spaces. For example, consider an RL agent to control an UAV that should capable of switching control modes between aggressive and conservative manners in different scenarios. Intuitively, attitude control is better modeled with continuous action space and the mode-switching is normally considered as a discrete action. Casting the mode switching into a continuous dimension results in difficulties in approximation and generalization; while discretizing continuous control space suffers from the scalability issue due to the exponentially exploring number of discretized action, which is also known as "curse of dimensionality". At the same time, many widely applied RL approaches have also been optimized to work with either discrete [13] or continuous action spaces [6, 14–16] but can rarely handle both. Although either approach can work in practice, in general, both strategies can weaken control authority or remove intrinsic structure from the problem, which can affect the policy performance or even make a problem harder to solve. Recent works such as Q-PAMDP [17] and Parameterized Deep Q-networks (P-DQN) [18] are proposed to combine a discrete and a continuous RL algorithm to tackle hybrid action spaces. The hybrid action space can be considered as a decision making upper layer and a continuous control lower layer, enabling the agent to react to the environment more flexibly.

Besides, designing a reward function that not only reflects the task property but is also elaborately shaped to guide the policy optimization is critical for RL applications. This can be reflected in many recent works [9, 10, 19, 20], where the authors design the reward function according to the specific problem, e.g. drone racing, obstacle avoiding and attitude control, etc. To design a well shaped reward function requires a lot of effort. Besides, shaped rewards may even hinder exploration at the initial stage of learning due to the penalization for inappropriate behaviors, therefore lead to sub-optimal learned policies. As a consequence, it is valuable to develop general RL algorithms which can learn from unshaped reward signals, e.g. a binary signal only indicating successful task completion. Hindsight experience replay

*PhD Student, Control and Simulation Division, Faculty of Aerospace Engineering, c.liu-10@tudelft.nl
†Assistant Professor, Control and Simulation Division, Faculty of Aerospace Engineering, e.vankampen@tudelft.nl

(HER) [21] is a technique that can be combined with RL algorithms to improve the sample efficiency with binary sparse rewards and therefore avoid the reward engineering. The effectiveness of HER is proved in many robotic applications with sparse rewards [7, 22].

Hybrid action space and sparse rewards encourage us to rethink problem definitions. In this paper, a UAV navigation task is formulated as a Parameterized Action Space Markov Decision Processes (PAMDP), a problem where the agent first selects a discrete action and subsequently a continuous set of parameters for that action, which can be considered as a discrete-continuous hybrid action space. In addition, a binary sparse reward setting that only indicates a successful task completion is considered. This setting intrinsically eliminates the possibly imperfect human-designed heuristics and saves human efforts in reward engineering. To tackle the hybrid action space and sparse reward challenges, a off-policy RL algorithm called HER-PDQN is proposed to treat the navigation problem in its native hybrid form, optimizes for discrete-continuous actions simultaneously with sparse rewards. The simulation results indicate that the proposed HER-PDQN algorithm has the promising ability to solve native hybrid reinforcement learning problems with less human efforts. The technical details of HER-PDQN is briefly explained in Section II.

II. Method

A. UAV Navigation as a PAMDP

a UAV navigation task is considered as a reinforcement learning problem with a discrete-continuous hybrid action space. Different from completely discrete or continuous actions that are widely studied in the existing literature, in our setting, the action is defined by the following hierarchical structure. We first choose a high level action k from a discrete set $[K]$ (we denote $1, ..., K$ by $[K]$ for short); upon choosing k, we further choose a low level parameter $x_k \in \mathcal{X}_k$ which is associated with the k-th high level action. Here $\mathcal{X}_k$ is a continuous set for all $k \in [K]$. Therefore, we get the discrete-continuous hybrid action space $\mathcal{A} = \{(k, x_k)|x_k \in \mathcal{X}_k \text{ for all } k \in [K]\}$. In our UAV navigation scenario, the high level discrete action set is either turn or move forward instructions. Given the high level discrete action, the low level angular acceleration to turn and acceleration power to move forward are equipped. The details of this environment is provided in section III.A. Based on the hybrid action space, the UAV navigation problem can be modeled as a Parameterized Action Markov Decision Process (PAMDP) [17]. For a PAMDP $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$: $\mathcal{S}$ is the set of all states, $\mathcal{A}$ is the parameterized (continuous) action space, $P(s'|s, k, x_k)$ is the Markov state transition probability function, $R(s, k, x_k, s')$ is the reward function, and $\gamma \in [0, 1)$ is the future reward discount factor [4]. An action policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ maps states to actions, typically with the aim of maximizing Q-values $Q(s, k, x_k)$, which give the expected discounted return of executing action (k, x_k) in state s and following the current policy thereafter.

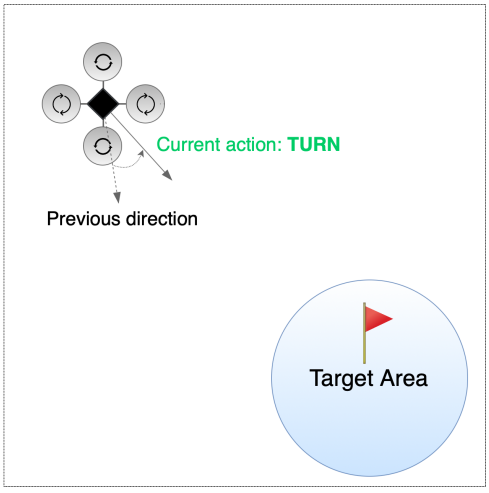

Fig. 1 The 2D navigation environment

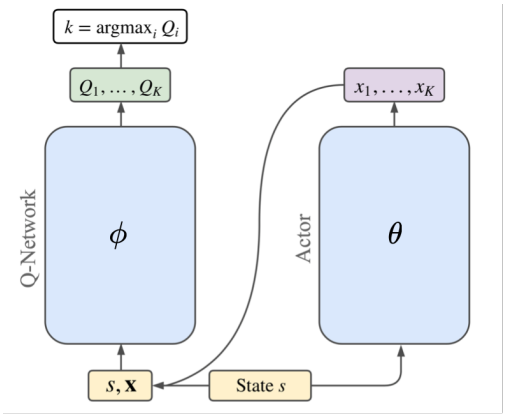

Fig. 2 The HER-PDQN architecture based on P-DQN[18]

B. Parameterized Deep Q-Networks in Hybrid Action Space

One of the state of the art reinforcement learning algorithm to deal with hybrid action spaces is Parameterized Deep Q-Networks(P-DQN) [18]. P-DQN uses a deep neural network with parameters ϕ to represent Q-values $Q_\phi(s, k, x_k)$ and a second deterministic actor network with parameters θ to represent the action-parameter policy $\pi_\theta : \mathcal{S} \rightarrow \mathcal{X}_k$. With this formulation it is easy to apply the standard DQN [13] approach of minimizing the mean-squared Bellman error $\mathcal{L}(\phi) = \frac{1}{N}\sum_{i=1}^{N}[y_i - Q_\phi(s_i, k, \pi_\theta)]^2$ to update the Q-network using N-sized mini-batches sampled from replay memory $\mathcal{D}$, where $y_i = r_i + \max_{k' \in [K]} \gamma Q_{\phi'}(s_{i+1}, k', \pi_{\theta'}(s_{i+1}))$ is the update target. Then, the loss for the actor network in P-DQN is given by the negative sum of Q-values: $\mathcal{J}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} Q_\phi(s_i, k, \pi_\theta(s_i))$ [18].

C. Hindsight Experience Replay

Hindsight Experience Replay (HER) [21] is another essential component that is integral to our method. HER helps agents learn policies much more efficiently when binary sparse rewards are used. The core idea behind HER is that even though an agent may have failed to achieve its given goal in an episode, the agent did learn a sequence of actions to achieve a different objective in hindsight — the state in which the agent finished. HER is implemented by creating a separate copy of the transitions $(s_t, a_t, r_t, s_{t+1}, g)$ that occurred in an episode where g represents the goal to be achieved, e.g. the desired target area for the navigation task. HER replaces (1) the original goal g with the goal achieved in hindsight g' and (2) the original reward r with the new value r' given g'. These transitions can increase the ratio of achieving goals successfully, and implicitly guide the agent to finish the original goal. To use HER, universal value function approximators (UVFA) [23] is needed to concatenate goals g with states s as a universal state $(s||g)$. Concretely, we use the *future* goal sampling strategy mentioned in [21] to sample m random states from the same episode as the transition being replayed and were observed *after* it. m is a hyperparameter that controls the ratio of HER data to data coming from normal experience replay in the replay buffer.

The architecture of HER-PDQN is shown in Fig. 2. For a clearer view, the detailed pseudocode for HER-PDQN is provided in algorithm 1. For E total training episodes, at each episode HER-PDQN will: (1) calculate the hybrid action and execute it to the environment at each timestep; (2) Get both original and HER transitions for each timestep and store them to the replay buffer; (3) Update neural network parameters for M steps.

III. Simulation Experiments and Results

A. Environment

A 2-dimensional navigation environment is created to evaluate our algorithm since there are no benchmark environments for hybrid RL with sparse rewards. Fig. 1 demonstrates our 2-dimensional UAV navigation environment. In order to navigate to the target area, the UAV agent need to firstly choose a discrete action between `turn` and `forward` based on its fully observation of current position state $s = (x, y)$ at each timestep, with (x, y) indicates the coordinate in the 2-dimensional state space. After the high-level action is chosen, Each action is then equipped with a low-level continuous parameter, indicating the acceleration power for moving forward or the angular acceleration for turning. The movement of the agent is always in the direction of its current direction. An episode ends if the agent stops in the target area (the winning state), it moves out of the field or the time limit exceeds.

B. Comparison between HER-PDQN and original P-DQN

In the experiments, we first trained P-DQN agent on the 2D navigation environment with hybrid action space and sparse rewards. We also trained our HER-PDQN agent on the same environment with $m = 4$ future goal sampling strategy, the comparison between HER-PDQN and P-DQN can be seen in Fig. 3. The timestep limit is set to 100 steps. At each time step, the agent receives a binary reward of -1. The agent receives a 0 reward only if it has reached the goal area. At the start of each episode, initial goals are randomly sampled from a uniform distribution. The randomly sampled initial goal stays fixed through the whole episode. We used Adam[24] optimizer with learning rate $\alpha_\phi = 10^{-2}$ and $\alpha_\theta = 10^{-3}$ for critic and actor networks in P-DQN respectively. The learning rate for critic is larger than actor because the critic provides the optimizing target for actor, as a consequence, although it is updating itself, the critic is supposed to converge faster than actor to provide a respectively stable and accurate guidance. exponential decay rates are set with $\beta_1 = 0.9, \beta_2 = 0.999$ and numerical stabilizer $\epsilon = 10^{-8}$ for Adam optimizer according to the default setting of [24]. We did independent training experiments for 10 runs with different random initialization of the networks. Each episode the UAV agent is initialized with the same position and orientation with 0 velocity. The goal which is also a

Algorithm 1: Parameterized Deep Q-Network with Hindsight Experience Replay (HER-PDQN)

Input: Greedy exploration parameter ϵ, mini-batch size N, hindsight experience replay ratio m, a uniform sample distribution $\mathcal{U}$ over action space $\mathcal{A}$, a gaussian exploration noise $\mathcal{N}$

Initialize actor network π_θ and critic network Q_ϕ with random weights θ and ϕ

Initialize target networks $\pi_{\theta'}$ and $Q_{\phi'}$ with weights $\theta' \leftarrow \theta, \phi' \leftarrow \phi$

Initialize replay buffer $\mathcal{D}$

for $episode = 1, E$ **do**

  Sample a goal g and an initial state s_0

  for $t = 0, T$ **do**

    Compute an continuous action a_t^c using the actor network π_θ and gaussian exploration noise $\mathcal{N}$:

    $a_t^c = \pi_\theta(s_t||g) + \mathcal{N}_t$

    Select discrete action a_t^d and hybrid action $a_t = (a_t^d||a_t^c)$ according to the ϵ-greedy policy:

    $$a_t = \begin{cases} \text{a random sample from distribution } \mathcal{U} & \text{with probability } \epsilon, \\ (\mathrm{a}_t^d||a_t^c) \text{ such that } a_t^d = \mathrm{argmax}_{k\in[K]} Q_\phi^k(s_t||g, a_t^c) & \text{with probability } 1-\epsilon. \end{cases}$$

    Execute action a_t, observe new state s_{t+1}

  end

  for $t = 0, T$ **do**

    $r_t := r(s_t, a_t, g)$

    Store the transition $(s_t||g, a_t, r_t, s_{t+1}||g)$ in $\mathcal{D}$

    Sample a m-sized set G of additional future goals for replay

    for $g' \in G$ **do**

      $r' := r(s_t, a_t, g')$

      Store the transition $(s_t||g', a_t, r', s_{t+1}||g')$ in $\mathcal{D}$

    end

  end

  for $iter = 1, M$ **do**

    Sample a batch of N transitions $\{s_i||g_i, a_i, r_i, s_{i+1}||g_{i+1}\}_{i\in[N]}$ randomly from $\mathcal{D}$

    Define the target $y_i = \begin{cases} r_i & \text{if } s_{i+1} \text{ is terminal,} \\ r_i + \max_{k\in[K]} \gamma Q_{\phi'}^k(s_{i+1}||g_{i+1}, \pi_{\theta'}(s_{i+1}||g_{i+1})) & \text{if otherwise.} \end{cases}$

    Optimize critic parameters ϕ by minimizing the loss: $\mathcal{L}(\phi) = \frac{1}{N}\sum_{i=1}^N [y_i - Q_\phi^{a_i^d}(s_i||g_i, a_i^c)]^2$

    Optimize actor parameters θ by minimizing the loss: $\mathcal{J}(\theta) = -\frac{1}{N}\sum_{i=1}^N \sum_{k=1}^K Q_\phi^k(s_i||g_i, \pi_\theta(s_i||g_i))$

    Update target network parameters:

    $$\phi' \leftarrow \tau\phi + (1-\tau)\phi'$$

    $$\theta' \leftarrow \tau\theta + (1-\tau)\theta'$$

  end

end

2-dimensional coordinate (x_g, y_g) is sampled from a uniform distribution within the boundaries of the environment. The number of training episodes is set to 200 since it can be observed that the policy will converge after around 100 episodes. The agent is optimized 40 times for each episode, in other words, parameter M in algorithm 1 is set to 40. This setting is in consistent with the implementation of original HER paper [21]. During optimization, we sample from the replay buffer with a batch size of 128 and do batch-wise update in parallel. The replay buffer size is set to be 50000.

Fig. 3 shows learning curves for HER-PDQN and P-DQN respectively. The curve is averaged on 10 independent runs with different random seeds, the shaded areas indicate the standard deviations from the mean value of returns. It can be seen that in this sparse reward environment, the original P-DQN performs poorly. Due to the sparse reward setting of the environment and the fact that P-DQN is not capable of dealing with sparse rewards, the agent struggles at learning effective policies that can fulfill this task. Specifically, P-DQN receives an averaged return around -100 even after 1000 epochs of learning, so the returns of P-DQN after 200 episodes are omitted for a clearer visual comparison with HER-PDQN. This phenomenon proved that human-designed reward function plays an essential role in reinforcement learning. However, it is clear that with the help of HER, the HER-PDQN can learn much more effectively and converged

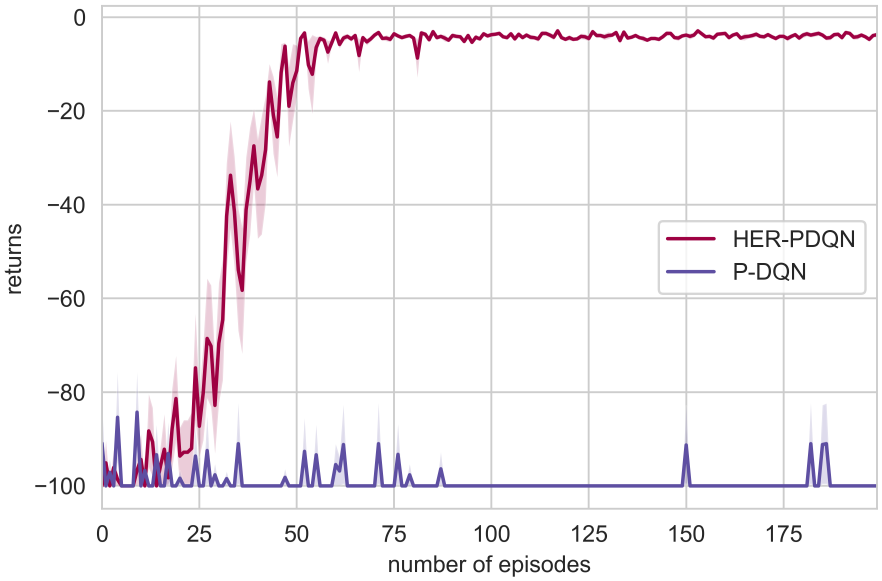

Fig. 3 Learning curves for HER-PDQN and P-DQN

to a stable policy around 50 episodes of training.

C. Future Goal Sampling Strategy

Note that from section II, the future goal sampling strategy is introduced to sample m random states from the same episode as the transition being replayed and were observed after it. While there are also other goal sampling strategies, it is shown in [21] that future sampling strategy performs the best. This makes sense because the future states encountered in the near future from the same episode has the largest potential to be achieved as a new goal. In this part, more experiments are implemented to study the effect of choosing different future sampling parameters m. m indicates the ratio of HER data to data coming from real experience replay interacting with the environment. We chose $m = 2, 4, 8, 32$ for each experiment, and train HER-PDQN with each m for 10 independent runs with different network random initialization. The other hyperparameters such as optimizer and network structure settings are the same.

Fig. 4 shows the learning curves of HER-PDQN with multiple future goal sampling parameters. It can be see that with $m = 2$ we get the worst performance, this is due to the smaller ratio of HER data to data coming from the real experience. A smaller m represents that the agent will receive less positive reward feedback from HER experiences. In other words, HER-PDQN degrades to original P-DQN when $m = 0$. From the comparisons between $m = 4, 8, 32$, it can be seen that these 3 agents performs nearly the same, even though $m = 32$ has the largest HER data percentage. As a consequence, $m = 4$ or 8 can be considered as ideal to generate enough HER experiences to facilitate learning process while requires less HER trajectory generation to save computation resources.

D. Visualization of Learned Hybrid Policy

Finally, the visualization of the hybrid policy for one sampled episode after training is provided. Fig. 5 shows how the HER-PDQN policy behaves after training. At the start of this episode, the initial goal is sampled to be positioned at the up-right corner while the UAV is initialized at the center of the environment facing straight up. The green arrows indicates the facing direction of the UAV, while the blue arrow represents the moving trajectory. Since the facing direction of UAV was not toward the goal initially, it performed a `turn` discrete action with the specific angle (yellow arrow) to face the goal area. Then it chose to move `forward` by accelerating towards the goal until it achieved the goal. The trained HER-PDQN agent has learned to turn towards its goal at first and move straightforward to the goal at full acceleration power, which is the optimal policy for this environment.

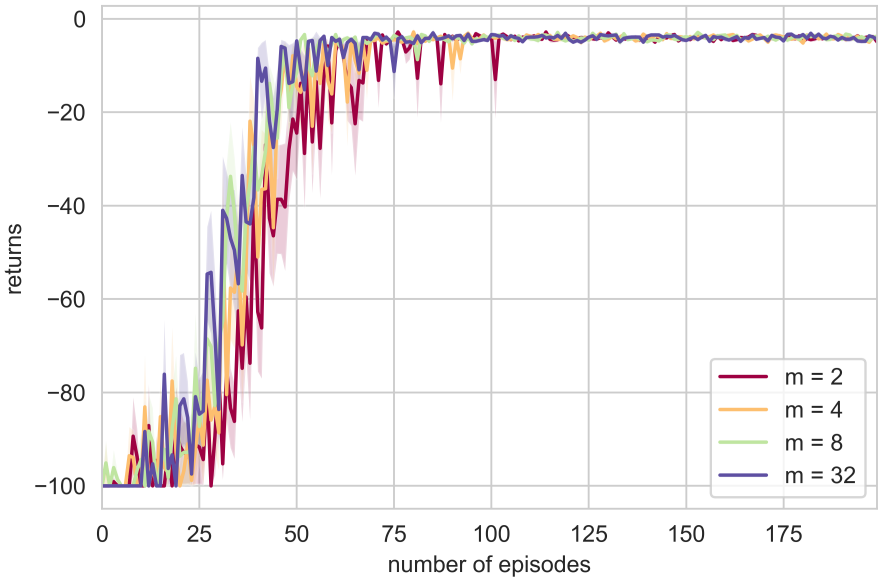

Fig. 4 Learning curves with different goal sampling parameter m

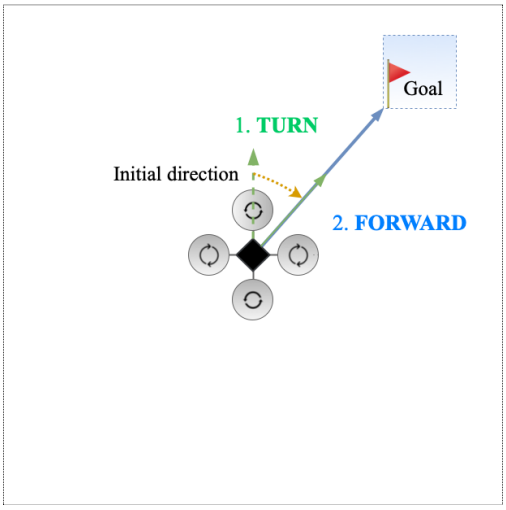

Fig. 5 Visualization of hybrid policy after training

IV. Conclusion

In this paper, a novel reinforcement learning algorithm called HER-PDQN is introduced, which combines Parameterized Deep Q-Networks with Hindsight Experience Replay to deal with a UAV navigation task with hybrid action spaces and sparse rewards. While one of the state of the art hybrid reinforcement learning algorithm fails on this challenging task, the proposed HER-PDQN algorithm can learn sample-efficiently and converge to stable policies that can fulfill the navigation task. HER-PDQN saves human effort by avoiding designing reward functions and alleviate the potentially biased human heuristics that may hinder the learning process. This RL approach can be utilized to design general UAV systems that deal with inherently hybrid action spaces where it is hard to design reward functions. Further research could be applying this algorithm to a real UAV and test in a more complex environment, for example with obstacles to avoid.

References

[1] Valenti, R. G., Jian, Y., Ni, K., and Xiao, J., "An autonomous flyer photographer," *2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2016, pp. 273–278.

[2] Valente, J., Cerro, J., Barrientos, A., and Sanz, D., "Aerial coverage optimization in precision agriculture management: A musical harmony inspired approach," *Computers and Electronics in Agriculture*, Vol. 99, 2013, pp. 153–159.

[3] Tomic, T., Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., Grixa, I. L., Ruess, F., Suppa, M., and Burschka, D., "Toward a Fully Autonomous UAV: Research Platform for Indoor and Outdoor Urban Search and Rescue," *IEEE Robotics Automation Magazine*, Vol. 19, No. 3, 2012, pp. 46–56.

[4] Sutton, R. S., and Barto, A. G., *Reinforcement Learning: An Introduction*, 2nd ed., The MIT Press, 2018. URL `http://incompleteideas.net/book/the-book-2nd.html`.

[5] Andrychowicz, M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J. W., Petron, A., Plappert, M., Powell, G., Ray, A., Schneider, J., Sidor, S., Tobin, J., Welinder, P., Weng, L., and Zaremba, W., "Learning dexterous in-hand manipulation," *Int. J. Robotics Res.*, Vol. 39, No. 1, 2020. https://doi.org/10.1177/0278364919887447, URL https://doi.org/10.1177/0278364919887447.

[6] Levine, S., Finn, C., Darrell, T., and Abbeel, P., "End-to-End Training of Deep Visuomotor Policies," *J. Mach. Learn. Res.*, Vol. 17, 2016, pp. 39:1–39:40. URL http://jmlr.org/papers/v17/15-522.html.

[7] Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P., "Sim-to-Real Transfer of Robotic Control with Dynamics Randomization." *ICRA*, IEEE, 2018, pp. 1–8. URL http://dblp.uni-trier.de/db/conf/icra/icra2018.html#PengAZA18.

[8] Gu, S., Holly, E., Lillicrap, T. P., and Levine, S., "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates." *ICRA*, IEEE, 2017, pp. 3389–3396. URL http://dblp.uni-trier.de/db/conf/icra/icra2017.html#GuHLL17.

[9] Hwangbo, J., Sa, I., Siegwart, R., and Hutter, M., "Control of a Quadrotor With Reinforcement Learning." *IEEE Robotics Autom. Lett.*, Vol. 2, No. 4, 2017, pp. 2096–2103. URL http://dblp.uni-trier.de/db/journals/ral/ral2.html#HwangboSSH17.

[10] Song, Y., Steinweg, M., Kaufmann, E., and Scaramuzza, D., "Autonomous Drone Racing with Deep Reinforcement Learning," *CoRR*, Vol. abs/2103.08624, 2021. URL https://arxiv.org/abs/2103.08624.

[11] Wang, C., Wang, J., Shen, Y., and Zhang, X., "Autonomous Navigation of UAVs in Large-Scale Complex Environments: A Deep Reinforcement Learning Approach," *IEEE Trans. Veh. Technol.*, Vol. 68, No. 3, 2019, pp. 2124–2136. https://doi.org/10.1109/TVT.2018.2890773, URL https://doi.org/10.1109/TVT.2018.2890773.

[12] Zhou, Y., van Kampen, E., and Chu, Q., "Incremental model based online heuristic dynamic programming for nonlinear adaptive tracking control with partial observability," *Aerospace Science and Technology*, Vol. 105, 2020. https://doi.org/10.1016/j.ast.2020.106013.

[13] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D., "Human-level control through deep reinforcement learning," *Nature*, Vol. 518, No. 7540, 2015, pp. 529–533. URL http://dx.doi.org/10.1038/nature14236.

[14] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., "Continuous control with deep reinforcement learning." *4th International Conference on Learning Representations, ICLR*, 2016. URL http://dblp.uni-trier.de/db/conf/iclr/iclr2016.html#LillicrapHPHETS15.

[15] Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P., "Trust Region Policy Optimization." *ICML*, JMLR Workshop and Conference Proceedings, Vol. 37, edited by F. R. Bach and D. M. Blei, JMLR.org, 2015, pp. 1889–1897. URL http://dblp.uni-trier.de/db/conf/icml/icml2015.html#SchulmanLAJM15.

[16] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O., "Proximal Policy Optimization Algorithms." *CoRR*, Vol. abs/1707.06347, 2017. URL http://dblp.uni-trier.de/db/journals/corr/corr1707.html#SchulmanWDRK17.

[17] Masson, W., Ranchod, P., and Konidaris, G. D., "Reinforcement Learning with Parameterized Actions," *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, edited by D. Schuurmans and M. P. Wellman, AAAI Press, 2016, pp. 1934–1940. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11981.

[18] Xiong, J., Wang, Q., Yang, Z., Sun, P., Han, L., Zheng, Y., Fu, H., Zhang, T., Liu, J., and Liu, H., "Parametrized Deep Q-Networks Learning: Reinforcement Learning with Discrete-Continuous Hybrid Action Space," *CoRR*, Vol. abs/1810.06394, 2018. URL http://arxiv.org/abs/1810.06394.

[19] Zhao, F., Zeng, Y., and Xu, B., "A Brain-Inspired Decision-Making Spiking Neural Network and Its Application in Unmanned Aerial Vehicle." *Frontiers Neurorobotics*, Vol. 2018, 2018. URL http://dblp.uni-trier.de/db/journals/finr/finr2018.html#ZhaoZX18.

[20] Hu, Z., Wan, K., Gao, X., Zhai, Y., and Wang, Q., "Deep Reinforcement Learning Approach with Multiple Experience Pools for UAV's Autonomous Motion Planning in Complex Unknown Environments," *Sensors*, Vol. 20, No. 7, 2020, p. 1890. https://doi.org/10.3390/s20071890, URL https://doi.org/10.3390/s20071890.

[21] Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W., "Hindsight Experience Replay," *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Curran Associates Inc., Red Hook, NY, USA, 2017, p. 5055–5065.

[22] Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P., "Overcoming Exploration in Reinforcement Learning with Demonstrations," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6292–6299. https://doi.org/10.1109/ICRA.2018.8463162.

[23] Schaul, T., Horgan, D., Gregor, K., and Silver, D., "Universal Value Function Approximators." *ICML*, JMLR Workshop and Conference Proceedings, Vol. 37, edited by F. R. Bach and D. M. Blei, JMLR.org, 2015, pp. 1312–1320. URL http://dblp.uni-trier.de/db/conf/icml/icml2015.html#SchaulHGS15.

[24] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.