ITC 1/53
Information Technology and Control
Vol. 53 / No. 1 / 2024
pp.237-242
DOI 10.5755/j01.itc.53.1.35135

Learning Stabilization Control of Quadrotor in Near-Ground Setting Using Reinforcement Learning

Received 2023/09/18 | Accepted after revision 2024/02/19

HOW TO CITE: Briliauskas, M. (2024). Learning Stabilization Control of Quadrotor in Near-Ground Setting Using Reinforcement Learning. *Information Technology and Control*, 53(1), 237-242. https://doi.org/10.5755/j01.itc.53.1.35135

Learning Stabilization Control of Quadrotor in Near-Ground Setting Using Reinforcement Learning

Mantas Briliauskas
Institute of data Science and Digital Technologies, Vilnius University,
Akademijos Str. 4, LT-08412, Vilnius, Lithuania; e-mail: mantas.briliauskas@mif.stud.vu.lt

Corresponding author: mantas.briliauskas@mif.stud.vu.lt

With the development of intelligent systems, the popularity of using micro aerial vehicles (MAV) increases significantly in the fields of rescue, photography, security, agriculture, and warfare. New modern solutions of machine learning like ChatGPT that are fine-tuned using reinforcement learning (RL) provides evidence of new trends in seeking general artificial intelligence. RL has already been proven to work as a flight controller for MAV performing better than Proportional Integral Derivative (PID)-based solutions. However, using negative Euclidean distance to the target point as the reward function is sufficient in obstacle-free spaces, e.g. in the air, but fails in special cases, e.g. when training near the ground. In this work, we address this issue by proposing a new reward function with early termination. It not only allows to successfully train Proximal Policy Optimization (PPO) algorithm to stabilize the quadrotor in the near-ground setting, but also achieves lower Euclidean distance error compared to the baseline setup.

KEYWORDS: quadrotor, stabilization, reinforcement learning, PPO, reward function for near ground flight.

1. Introduction

Over the recent decade the popularity of quadrotor MAVs has been increasing in various fields both in indoor and outdoor environments due to their simplicity in design, small size, great dynamical maneuverability, and operations in 3D [3]. Quadrotor is controlled by providing input to each of four rotors and its state is changed with respect to six degrees of freedom. Due to this fact, the quadrotor is an underactuated system.

Maneuvering the quadrotor is a non-linear problem. It is a challenging task for humans and only experienced operators are able to control MAV without an

additional stabilization [21]. Consequently, quadrotor control assistance is required to address the specialized MAV personnel demand issue.

For quadrotor stabilization, linear methods as PID, Linear Quadratic Regulator (LQR), and non-linear like Nonlinear Hierarchical Control Strategy (NHCS), Back-Stepping (BS), Dynamic Surface Control (DSC) are used [9]. However, those methods are not robust to system uncertainties or external disturbances [9]. Additionally, in case of cargo-carrying MAVs the model dynamics are changed with every different load [2]. Popular methods like PID-based control [7] require provision of their explicit parameters or comprehensive parameters tuning [23, 5]. Deep reinforcement learning (DRL) appears to be a good candidate to address these issues [9] by providing a practical solution without an explicit need for knowledge about quadrotor dynamics, motion model and MAV control.

To compare with ground-based robots, MAVs operate in 6 degrees of freedom, use higher velocities, therefore, are more prone to damage. This makes practical experiments expensive. Therefore, model fitting and testing of the maneuvering algorithms are performed under a simulated environment.

Despite DRL being capable of solving the stabilization problem, adding obstacles to the environment requires behavior correction, so reward engineering [7] is required. A solution to the stabilization problem when the obstacle lays under MAV like ground is the main outcome of this paper.

This paper begins with the latest literature review in MAV stabilization topic, then follows the section describing reinforcement learning approach and the method that is chosen in this work. Next, the details about experiment setup are provided and the section with the proposed approach of this paper. After that, the results of the proposed approach are compared to results of the baseline approach, following the discussion section.

2. Related Works

Due to internal system uncertainties and external disturbances, the quadrotor might drift away from its set-point. The stabilization system task is to keep tracking the quadrotor's state and, if necessary, navigate it back to the setpoint. This process permits to frame the quadrotor stabilization problem as a type of quadrotor navigation problem where the next target is constantly a set-point.

Prior works addressing stabilization problem using RL can be categorized into those using additional methods like PID and those that are purely-based on RL.

In 2016, Sugimoto and Gouko [21] took a Q-learning approach to stabilize the quadrotor over a particular point and showed that it performs as well as a PID controller. Alrubyli and Bonarini [1] approached to stabilize the UAV in the z-axis by applying Q-learning for PID controller. Koch et al. [7] showed that training PPO algorithm for unmanned aerial vehicle (UAV) attitude control outperforms PID controller used in Betaflight software. These attempts show that RL can perform no worse than PID controller. The first one used 3x3 reward grid with highest reward at the center to stabilize quadrotor at particular position, the second used negative next-timeframe altitude distance from the set-point as a reward function and the third normalized angular velocity error as a reward function to stabilize quadrotor at target attitude.

Hwangbo et al. [4] in use a deterministic on-policy method to learn quadrotor stabilization by using zero-bias and zero-variance samples. They demonstrate that their learned policy performs two orders of magnitude faster than the common trajectory optimization algorithms. Instead of reward, they used a cost function and minimized the sum of position, angular velocity and linear velocities. Olaz et al. [13] used PPO and DDPG for stabilized taking-off and landing of the UAV in the windy environment. They made a decoupling by outputting target forces and momentum from the actor neural network and then linearly transforming those to motor angular velocities. Jiang and Lynch [6] trained hover stabilization policy using PPO considering full dynamics of quadrotor in model-free setting. They emitted a reward if the quadrotor is proximal to set-point and tested three reward settings - using velocity-only, by adding position shift, and adding time as penalty. They show that such a policy can perform no worse than manually tuned PD controller. Lin et al. in [9] proved that RL-based actor-critic policy can be employed to mitigate external disturbances like wind and demonstrated it in simulator experiments. They used a reward system including position, velocity and control inputs to the

engines. The set-point tracking can also be achieved using on-boarding actor-critic evaluated in Field Programmable Gate Arrays (FGPA) for energy efficient real-time control. Li et al. [8] in their work show that their approach achieves 58.14% less position error compared to PID approach. They used transformed Manhattan distance between the desired pose and the actual pose as a reward function. Dooraki and Lee [16] used a bio-inspired solution to replace conventional controller and implemented a PPO model-free flight controller that tracked the set-point. In their work, they did not explicitly provide the reward function, but stated that it was based on positions and attitudes, also added limits for pitch and roll angles considering them as terminal states, and added negative reward for bypassing limits of x, y, z components.

The review shows that a special case of near-ground setting has not yet been addressed.

3. Reinforcement Learning

Reinforcement learning is the third branch of machine learning after supervised and unsupervised learning. It increasingly gained popularity with a rise of neural networks as Google Brain using deep Q-network [12] taught agent to play Atari games from pixels, played games in human master level: GO [19] and chess [20].

The main distinction between reinforcement learning compared to supervised and unsupervised learning is learning from interaction [22]. The agent interacts with the environment by making actions and receives next observation and reward. Learning is done in episodes and one episode is described by trajectory. The goal of RL is to generate an interaction strategy called policy such that would maximize the cumulative reward over time. An accumulative reward of one episode is also called return. The expected reward starting from the current state is called value. Its calculation is one of the principal RL problems, since having a value function would be sufficient to receive maximum return [22]. Worth to mention, the agent is usually forced to collect a reward as early as possible and for that matter, a discount rate is added to the return function.

Formally, RL is framed as Markov Decision Process (MDP). It inherits Markov property, so the state and the reward are only dependent on the previous state and the action taken. The process of taking an action may be framed as a stochastic process, in that case the next observation is a random variable defined by the transition function.

With the rise of neural networks, policy gradient methods gained popularity as they are capable of working in stochastic action selection, and also supports continuous action spaces. This expands their applicability, moreover implicitly addresses the exploration-exploitation dilemma. Apart from that, policy gradient methods tend to suffer from high variance. To mitigate it, various variance reduction techniques were applied. One successful approach is taken in the actor-critic framework. In this setup, the critic learns the value of the action given a state, and that value is compared with the actual return collected by the actor. It gives an advantage setting measuring how a better actor performs compared to previous iterations and so is called an advantage actor critic (A2C) framework [11].

Another important issue of reinforcement learning is drift off the learnt. It is common when a slight change in policy completely changes the trajectory distribution, and consequently the actor falls into low-reward trajectories space and its policy degrades. To mitigate this issue, various distribution change control techniques are proposed. Few successful approaches were Trust Policy Region Optimization (TRPO) [17] and PPO [18]. The TRPO aims to limit change in action distribution by KL-divergence constraint, PPO clips the change of distribution that is higher than the set ratio. The latter was most popular among reviewed papers [7, 6, 16, 10], therefore it is employed as a policy training algorithm in this paper.

Besides optimal policy search, shaping of the reward function is important to instruct agent which states and/or states are expected by providing high reward and which not by providing low reward. In this work, the importance of reward function is addressed and shown that the naive form of reward function can lead to an unlearned algorithm and an improvement to that function is proposed.

4. Experiment Setup

Bullet simulator was used within the gym-pybullet-drones library [14]. For the quadrotor model, the Bitcraze's Crazyflie 2.x nano-quadrotor was chosen

that has motor-to-motor dimensions of 92x92x-29mm, it is by default implemented in the gym-pybullet-drones library. The simple drag, ground effect, and down-wash models are present in the gym-pybullet-drones library.

The agent had an observation consisting of position, orientation and difference in position and orientation from the last time step, so o ∈ R12. Action space is continuous, it corresponds to speed in revolutions per minute (rpm) for each rotor, a ∈ [0; 21702.64].

In each episode the quadrotor set-point (stabilization position) was kept constant [1;1;1]. The starting position was taken randomly x, y ∈ [−1, 1] to make sure agent learns to achieve set-point from different positions, starting altitude was kept constant z = 0.1 in order to begin near ground.

For the RL part, the advantage of actor-critic architecture was used with PPO training solution.

For the baseline setup the reward function was negative Euclidean distance from the quadrotor to the set-point:

$$r_{baseline} = -\|p_{quad} - p_{target}\|, \quad (1)$$

where p_{quad} is quadrotor's position, p_{target} is a set-point coordinate, $\|\cdot\|$- vector's l^2-norm.

Each reward function was used during training PPO from stable baselines3 python library [15] with default parameters. Maximum length of one episode was 10 seconds, which is equivalent to 482 time steps in a simulator. If the quadrotor hit the ground, the episode was set to terminate.

Goal of stabilization is to minimize the distance from the quadrotor to the setpoint. In order to compare different reward functions, the average Euclidean distance per episode as a metric for comparison is selected.

5. Proposed Reward Function

In this work the composite reward function with early termination is proposed.

$$r_{proposed} = r_p + r_g, \quad (2)$$

where r_p is a reward based on quadrotor position, r_g is a negative reward values by constant g_{pen} when quadrotor's altitude is lower than a:

$$r_g = \begin{cases} g_{pen}, & if\ p_{quad,z} < a \\ 0, & otherwise \end{cases} \quad (3)$$

$$r_p = -min(\|p_{quad} - p_{target}\|, d_{lim}), \quad (4)$$

where d_{lim} is a limit for Euclidean distance error. The $p_{quad,z} < a$ condition indicates quadrotor reached the ground and the episode is terminated.

6. Results

Two reward functions were used - the baseline $r_{baseline}$ and the proposed $r_{proposed}$ with early termination. Each iteration the PPO algorithm was trained 3 million time steps and 5 times, with random starting position and random initial neural network parameters of actor-critic. The results are shown below in the plots, the 5-run mean is plotted along with confidence interval of 1 standard deviation.

The experiment shows that using the baseline reward function $r_{baseline}$ it was not able to train the PPO model to stabilize the quadrotor at the set-point (see figure 1). The distance of around 0.896 indicates the average quadrotor position from the set-point during the episode. According to the fact that the start point is approximately 1 meter away from the set-point (see figure 2), the quadrotor was not able to navigate to the set-point in general. This means failure in learning and shows that in this setup the baseline reward function is not suitable to train PPO to stabilize the quadrotor in the near-ground setting.

Figure 1

The mean episode return (accumulative reward) during PPO training using baseline reward function $r_{baseline}$

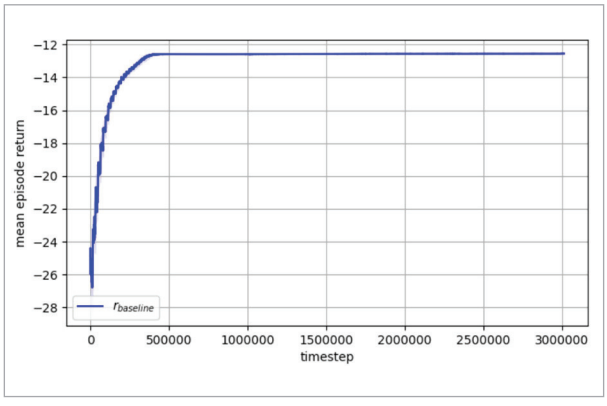
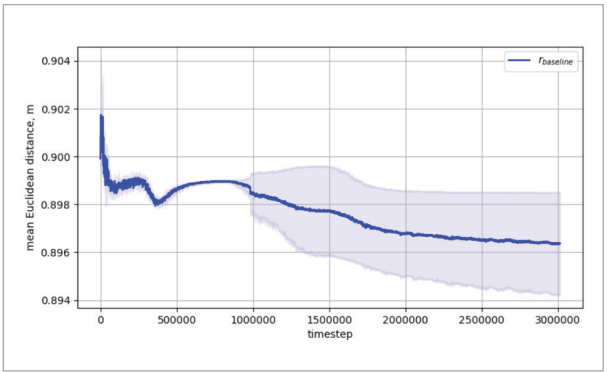


Figure 2
Mean Euclidean distance from quadrotor to set-point throughout the episode during training of PPO using baseline reward function $r_{baseline}$. The results show that the mean episode distance does not fall below 0.894, showing that the model used not to reach the target and the learning failed



On the contrast, using the proposed reward function $r_{proposed}$, the average distance from the set-point during the episode was approximately 0.15 meter (see figure 3). The algorithm converges in approximately 2.7-3M

Figure 3
Mean Euclidean distance from quadrotor to set-point throughout the episode during training of PPO using proposed reward function $r_{proposed}$. The mean distance plateaus around 0.15m, which indicates the time quadrotor has raising up to the target and stabilizes

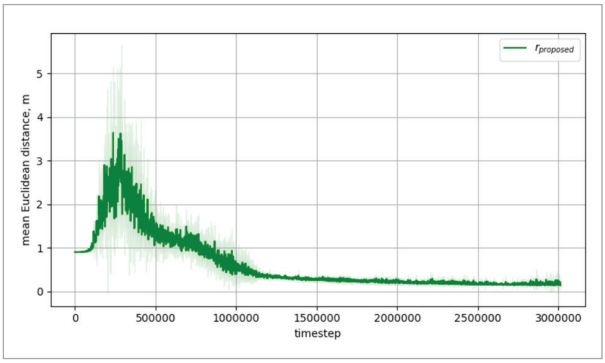
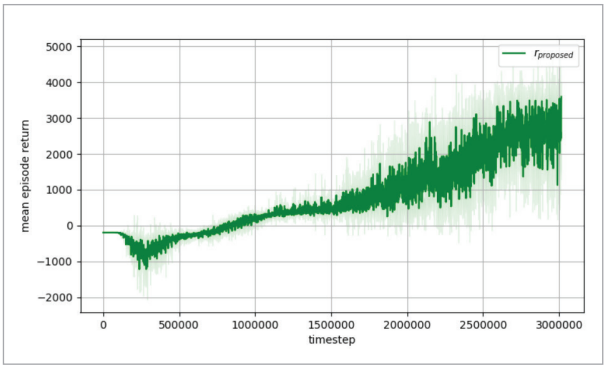


Figure 4
The mean episode return (accumulative reward) during PPO training using the proposed reward function $r_{proposed}$ with early termination



timesteps (see figure 4). Since the episode was taking 10 seconds, this number means the quadrotor was launching up from the current state, reached the target and stabilized.

7. Conclusions

Training reinforcement learning algorithm for quadrotor control is not an easy task, even using state-of-the-art algorithms. The importance of the reward function with early termination was stressed in this work.

In this work it was shown that using a proposed reward function with early termination gives more than 4x shorter average distance from set-point in near-ground setup compared to negative Euclidean distance function. This approach will help to advance the research topic of reinforcement learning in quadrotor control by addressing the issue of policy optimization falling into local minima in the near-ground setting. To conclude, using the suggested approach the quadrotor reaches the stabilization point within 1.2M timesteps on average, whereas the baseline method does not reach the stabilization point within 2.7-3M timesteps.

References

1. Alrubyli, Y., Bonarini, A. Using Q-learning to Automatically Tune Quadcopter PID Controller Online for Fast Altitude Stabilization. IEEE International Conference on Mechatronics and Automation (ICMA), 2022, 514-519. https://doi.org/10.1109/ICMA54519.2022.9856292
2. Eduardo, G. S., Caarls, W. Designing a Robust Low-Level Agnostic Controller for a Quadrotor with Actor-Critic Reinforcement Learning, 2022. arXiv:2210.02964. https://doi.org/10.1109/ICMA54519.2022.9856292

3. Emran, B. J., Najjaran, H. A review of Quadrotor: an Underactuated Mechanical System. Annual Reviews in Control, 2018, 46, 165-180. https://doi.org/10.1016/j.arcontrol.2018.10.009

4. Hwangbo, J., Sa, I., Siegwart, R., Hutter, M. Control of a Quadrotor With Reinforcement Learning. IEEE Robotics and Automation Letters, 2017, 2(4), 2096-2103. https://doi.org/10.1109/LRA.2017.2720851

5. Javeed, A., Jiménez, V. L., Grönqvist, J. Reinforcement Learning-Based Control of CrazyFlie 2.X Quadrotor, 2023. arXiv:2306.03951. https://doi.org/10.1109/LRA.2017.2720851

6. Jiang, Z., Lynch, A. F. Quadrotor Motion Control Using Deep Reinforcement Learning. Journal of Unmanned Vehicle Systems, 2021, 9(4), 234-251. https://doi.org/10.1139/juvs-2021-0010

7. Koch, W., Mancuso, R., West, R., Bestavros, A. Reinforcement Learning for UAV Attitude Control, 2018. arXiv:1804.04154.https://doi.org/10.1139/juvs-2021-0010

8. Li, Y., Li, H., Li, Z., Fang, H., Amit, S., Wang, Y., Qiu, Q. Fast and Accurate Trajectory Tracking for Unmanned Aerial Vehicles based on Deep Reinforcement Learning. 2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), 2019, 1-9. https://doi.org/10.1109/RTCSA.2019.8864571

9. Lin, X., Yu, Y., Sun, C. Supplementary Reinforcement Learning Controller Designed for Quadrotor UAVs. IEEE Access, 2019, 7, 26422-26431. https://doi.org/10.1109/ACCESS.2019.2901295

10. Lopes, G., Ferreira, M., Simões, A., Colombini, E. Intelligent Control of a Quadrotor with Proximal Policy Optimization Reinforcement Learning. 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE), 2018, 503-508. https://doi.org/10.1109/LARS/SBR/WRE.2018.00094

11. Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., Kavukcuoglu, K. Asynchronous Methods for Deep Reinforcement Learning, 2016. arXiv:1602.01783. https://doi.org/10.1109/LARS/SBR/WRE.2018.00094

12. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M. Playing Atari with Deep Reinforcement Learning, 2013. arXiv:1312.5602. https://doi.org/10.1109/LARS/SBR/WRE.2018.00094

13. Olaz, X., Alaez, D., Prieto, M., Villadangos, J., Astrain, J. J. Quadcopter Neural Controller for Take-off and Landing in Windy Environments. Expert Systems with Applications, 2023, 225, 120146. https://doi.org/10.1016/j.eswa.2023.120146 https://www.sciencedirect.com/science/article/pii/S0957417423006486

14. Panerati, J., Zheng, H., Zhou, S., Xu, J., Prorok, A., Schoellig, A. P. Learning to Fly-a Gym Environment with PyBullet Physics for Reinforcement Learning of Multi-agent Quadcopter Control. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021. https://doi.org/10.1109/IROS51168.2021.9635857 https://doi.org/10.1109/LARS/SBR/WRE.2018.00094

15. Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., Dormann, N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. Journal of Machine Learning Research, 2021, 22(268), 1-8. https://doi.org/10.1109/LARS/SBR/WRE.2018.00094

16. Ramezani Dooraki, A., Lee, D.-J. An Innovative Bio-inspired Flight Controller for Quad-rotor Drones: Quad-rotor Drone Learning to Fly Using Reinforcement Learning. Robotics and Autonomous Systems, 2021, 135, 103671. https://doi.org/10.1016/j.robot.2020.103671

17. Schulman, J., Levine, S., Moritz, P., Jordan, M. I., Abbeel, P. Trust Region Policy Optimization, 2017. arXiv:1502.05477. https://doi.org/10.1109/LARS/SBR/WRE.2018.00094

18. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O. Proximal Policy Optimization Algorithms, 2017. arXiv:1707.06347. https://doi.org/10.1109/LARS/SBR/WRE.2018.00094

19. Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D. Mastering the Game of Go with Deep Neural Networks and Tree Search. Nature, 2016, 529(7587), 484-489. https://doi.org/10.1038/nature16961

20. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., Hassabis, D. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm, 2017. arXiv:1712.01815. https://doi.org/10.1038/nature16961

21. Sugimoto, T., Gouko, M. Acquisition of Hovering by Actual UAV Using Reinforcement Learning. 3rd International Conference on Information Science and Control Engineering (ICISCE), 2016, 148-152. https://doi.org/10.1109/ICISCE.2016.42

22. Sutton, R. S., Barto, A. G. Reinforcement learning: An Introduction, 2018.https://doi.org/10.1109/ICISCE.2016.42

23. Yoon, J., Doh, J. Optimal PID Control for Hovering Stabilization of Quadcopter Using Long Short Term Memory. Advanced Engineering Informatics, 2022, 53, 101679.https://doi.org/10.1016/j.aei.2022.101679