Jurnal Minfo Polgan
Volume 13, Nomor 1, 2024
DOI : https://doi.org/10.33395/jmp.v13i1.13489
e-ISSN : 2797-3298
p-ISSN : 2089-9424

Terbit : 13 Februari 2024

Utilizing Deep Reinforcement Learning, EGO-Swarm Parameter Control determines the path of the drone.

[1] Conrad Bombongan, ST, MT, [2] Tanda Selamat, [3] Johan, [4] Waisen
[1, 2,3,4] Universitas IBBI, Indonesia

[1] conrad_nainggolan@yahoo.com , [2] tandaselamat@gmail.com , [3] joh4nhu4ng@gmail.com,
[4] whisen@gmail.com

ABSTRAC

The recent attention garnered by the EGO-Swarm system is due to its successful generation of smooth drone flight paths in complex real-world environments. Among the various parameter values utilized by EGO-Swarm for path generation, the maximum speed and maximum acceleration of the drone play a critical role in enhancing flight performance. However, in the current EGO-Swarm setup, these parameter values remain fixed during path creation, despite the potential need for real-time adjustments in dynamic environments. Therefore, there is a need to dynamically adapt these values to the evolving real-time environment. This paper proposes a novel algorithm that dynamically adjusts the maximum speed and maximum acceleration using hierarchical deep reinforcement learning in response to real-time environmental changes. To assess the effectiveness of the proposed method, a comparative experiment is conducted between the existing EGO-Swarm algorithm and the proposed algorithm in a ROS simulation. The experimental results reveal that the proposed algorithm surpasses the existing EGO-Swarm algorithm in terms of average speed improvement and path length.(Zhou et al., 2021)

Keyword: Drone Autonomous Flight, Hierarchical Deep Reinforcement Learning, EGO-Swarm.

INTRODUCTION

In light of the rising drone usage across diverse domains, there's been a surge in research on autonomous drone navigation. In such research, it's imperative to concurrently address dynamics, path smoothness, and safety to generate real-time paths that evade obstacles and mitigate drone mechanical issues. These requirements for autonomous drone navigation become even more critical in complex environments but pose significant challenges to resolve. (Yang et al., 2023)

EGO-Swarm, an algorithm for drone autonomous driving, is a system equipped with algorithms that generate real-time paths for multiple drones. Notably, its ability to create real-time paths for drones in complex real-world environments has garnered considerable interest from researchers in related fields. EGO-Swarm encompasses numerous parameters for the creation of smooth paths, including the maximum speed and maximum acceleration of the drone. However, while the optimal values for maximum speed and maximum acceleration vary with changes in the real-time environment where the current path is generated, EGO-Swarm maintains the initially set values. (Park et al., 2022)

Hence, this paper employs hierarchical deep reinforcement learning to determine the maximum speed and maximum acceleration based on real-time environmental changes. The upper-layer agent utilizes the Soft-Actor-Critic (SAC) algorithm to dynamically set the maximum speed and maximum acceleration while considering the evolving environment. Meanwhile, the lower-layer agent utilizes the set maximum speed and maximum acceleration to generate a real-time route using the EGO-Swarm algorithm. Through a comparative analysis conducted in a ROS simulation representing a complex environment, we demonstrate the superiority of the proposed algorithm over the existing EGO-Swarm algorithm in terms of speed, path length, and path smoothness.

Jurnal Minfo Polgan
Volume 13, Nomor 1, 2024
DOI : https://doi.org/10.33395/jmp.v13i1.13489
e-ISSN : 2797-3298
p-ISSN : 2089-9424

LITERATURE REVIEW

EGO-Swarm functions as a path creation algorithm for enabling the autonomous navigation of multiple drones, known as swarm drones, in complex environments like forests, devoid of prior information. This algorithm constructs a route solely based on onboard drone data, without relying on external location information or pre-existing knowledge of the driving environment. (Nguyen et al., 2021)

EGO-Swarm devises a route by concurrently optimizing time and space while adjusting the temporal aspect of each drone. This capability enables EGO-Swarm to generate paths within milliseconds, even in intricate environments devoid of prior information. Moreover, EGO-Swarm can incorporate specific missions, such as object tracking and formation maintenance, through a multi-objective optimization function. It facilitates swarm drones to share trajectories, thereby minimizing data transmission, even in unreliable communication networks. (Alyassi et al., 2022)

RESEARCH METHODS

Reinforcement learning, a subset of machine learning, involves an agent learning to make optimal decisions through interaction with its environment. The agent learns by maximizing the rewards obtained from the actions it takes. Soft-Actor-Critic (SAC), a reinforcement learning algorithm, builds upon the Actor-Critic algorithm framework, comprising Actor and Critic components. (Haarnoja, Zhou, Hartikainen, et al., 2018) The SAC algorithm introduces an entropy term to the objective, which guides actor learning, promoting exploration. Maximizing entropy enables consideration of not only the optimal policy defined by the current actor but also alternative approximate optimal policies. This enhances the effectiveness of exploration and facilitates the discovery of diverse approximate optimal policies, ensuring robust learning. (Haarnoja, Zhou, Abbeel, et al., 2018)

As an off-policy algorithm, the SAC algorithm offers the advantage of efficient learning data sampling, as it allows for the reuse of utilized learning data. In the SAC algorithm, the objective of the Critic is defined as follows. (de Jesus et al., 2021)

$$J_Q(\theta) = E_{(s_t,a_t)\sim D}[\frac{1}{2}(Q_\theta(s_t,a_t) - (r(s_t,a_t) + \gamma(Q_{\bar{\theta}}((s_{t+1},a_{t+1}) - \alpha\log(\pi_\phi(a_{t+1}|s_{t+1}))))^2]$$

In this context, Q represents the state-action value, while r denotes the reward function of the reinforcement learning environment. Additionally, a and s refer to action and state, respectively. Lastly, π defines the Actor, and when a specific s is provided as input, learning is conducted with the aim of extracting the optimal a.

Hierarchical deep reinforcement learning

Hierarchical deep reinforcement learning entails a framework within deep reinforcement learning that addresses multiple levels of sub-problems to solve the overarching problem. Typically, the upper-layer agent in hierarchical deep reinforcement learning categorizes problems broadly and selects fundamental actions, while the lower-layer agent learns to tackle sub-problems arising from the actions chosen by the upper-layer agent. Within hierarchical deep reinforcement learning, two agents in distinct layers operate at varying levels of time-step horizons, with the upper agent functioning at a higher time scale and the lower agent at a lower time scale. Hierarchical deep reinforcement learning offers the advantage of decomposing complex tasks into a hierarchical structure of sub-problems, facilitating relatively straightforward generalization and high learning efficiency. (Duan et al., 2021)

EGO-Swarm encompasses various parameters for generating a smooth path, including the maximum speed and maximum acceleration of the drone. The optimal values for these parameters fluctuate based on the drone's immediate surroundings where the current path is being generated. However, the current EGO-Swarm implementation maintains the initial settings for maximum speed and maximum acceleration without adapting to changes in the real-time environment.

To address this limitation, this paper introduces an algorithm that dynamically adjusts the maximum speed and maximum acceleration to optimal values considering the evolving environment during

Jurnal Minfo Polgan
Volume 13, Nomor 1, 2024
DOI : https://doi.org/10.33395/jmp.v13i1.13489
e-ISSN : 2797-3298
p-ISSN : 2089-9424

path creation. The proposed algorithm leverages a hierarchical deep reinforcement learning framework and demonstrates enhancements in drone speed, path smoothness, and path length compared to the existing EGO-Swarm approach. (Wayahdi et al., 2021)

Proposed hierarchical deep reinforcement learning framework

In this study, we employ hierarchical deep reinforcement learning, as introduced, to adjust the real-time parameters of EGO-Swarm. The proposed algorithm, which utilizes hierarchical deep reinforcement learning, involves the upper-layer agent employing the deep reinforcement learning SAC algorithm to compute the drone's maximum speed and maximum acceleration, taking into account the dynamically changing current environment. Subsequently, utilizing these parameters, the EGO-Swarm algorithm in the lower layer generates a path for autonomous drone flight. In essence, our approach entails leveraging deep reinforcement learning in the upper layer and employing the traditional control algorithm presented by EGO-Swarm in the lower layer.

Algorithm 1: Proposed Algorithm

G: global target
O: environment surrounding the drone
P: position of drone
π_ϕ: actor of SAC
$\mathcal{Q}_\theta$: critic of SAC (state action value function)
$\mathcal{B}$: replay memory

inialize $\mathcal{B}$
Set G
repeat
  foreach *each episode* **do**
    foreach *each episode step* **do**
      $MaxVel_t, MaxAcc_t \sim \pi_\phi(O_t)$; // high level agent
      /* low level agent starts */
      $\Gamma = \text{EgoSwarm}(MaxVel_t, MaxAcc_t, G, O_t)$
      foreach *t second* **do**
        $P \sim MoveDroneAlongTraj(\Gamma)$
        CheckCollision(P)
      /* low level agent ends */
      $O_{t+1} \sim DroneSensor(P)$
      $\mathcal{B} \leftarrow \mathcal{B} \cup (O_t, (MaxVel_t, MaxAcc_t), r_t, O_{t+1})$
    foreach *the number of training* **do**
      Training SAC agent (i.e. π_ϕ and $\mathcal{Q}_\theta$)
until *Drone Reach* G;

Figure 1 Proposed algorithm pseudocode

The state, a component of the upper layer Markov Decision Process (MDP) utilizing the deep reinforcement learning SAC algorithm, comprises surrounding obstacles detected by the drone's sensor and directional information to the destination. Meanwhile, the action, situated in the lower layer, pertains to determining the maximum speed and maximum acceleration of the drone utilized in the EGO-Swarm algorithm. Rewards are assigned as follows: +1.0 upon reaching the destination, -1.0 upon colliding with an obstacle, and -0.01 in other states. The designated MDP is not impulsive but rather fosters learning aimed at achieving the destination at an accelerated pace.

In the interim, following the creation of the drone path in the lower layer, the drone traverses the path for several seconds, while one SAC agent time step is executed in the upper layer. The episode of the SAC algorithm in the upper layer concludes when the drone encounters an obstacle or reaches its destination in the lower layer. (Hafiz et al., n.d.)

The pseudocode outlining the proposed algorithm is depicted in [Figure 1]. Initially, the drone's ultimate destination is set, and the environmental data detected by the drone's sensor at its current location is provided as input to the upper layer agent. Subsequently, the upper layer agent processes the surrounding environmental information. Once the maximum speed and maximum acceleration of the drone are factored in, the lower layer agent is initiated. This lower-layer agent then generates a route within EGO-Swarm, considering the drone's maximum speed and maximum acceleration,

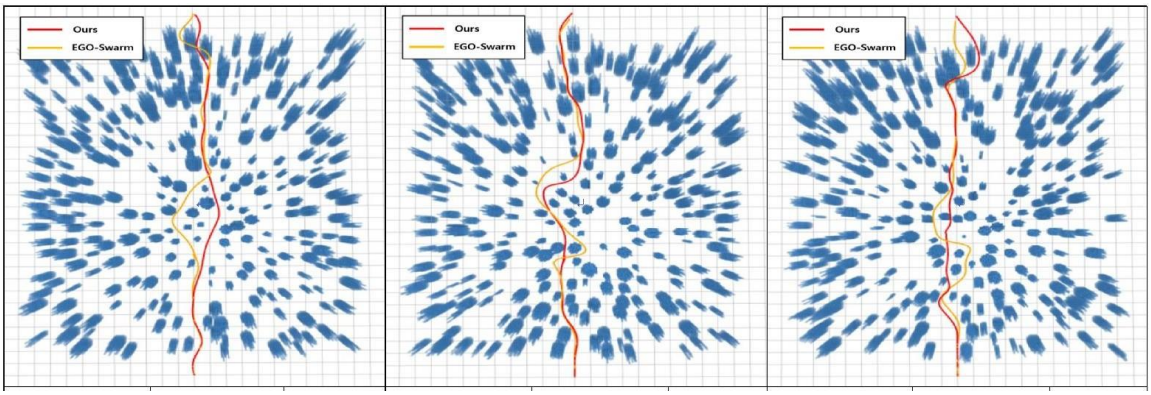
Jurnal Minfo Polgan
Volume 13, Nomor 1, 2024
DOI : https://doi.org/10.33395/jmp.v13i1.13489
e-ISSN : 2797-3298
p-ISSN : 2089-9424

final destination, and real-time environmental data obtained from the upper layer. The drone subsequently traverses along the designated path for a specified duration. At the conclusion of each episode, the SAC algorithm, operating as the higher layer agent, undergoes training. These series of procedures are iterated until the drone successfully reaches its destination.

RESULTS AND DISCUSSION

In this study, we perform a comparative experiment utilizing the ROS-based simulation offered by EGO-Swarm. EGO-Swarm facilitates ROS simulation of intricate environments. The comparative experiments were carried out across a total of three environments. In each environment, drones navigate from the starting point to the destination following the path generated by both EGO-Swarm and the proposed algorithm.

In the experiment results depicted in [Figure 2], the red path represents the path generated using the proposed algorithm, while the orange path corresponds to the path generated using EGO-Swarm. It is evident that the proposed algorithm exhibits a higher average speed than EGO-Swarm in each environment. Consequently, owing to the higher average speed, the time taken to reach the destination is reduced. Furthermore, as illustrated in the figure, the overall smoothness of the path is notably enhanced in the proposed algorithm. Consequently, there is a decrease in the path length required to reach the destination.



Performance comparison experiment results between the proposed algorithm and the EGO-Swarm algorithm in the ROS simulation environment are depicted in Figure 2.

The experimental results indicate that the proposed algorithm produces routes with higher average speeds and smoother paths compared to EGO-Swarm across various ROS simulation environments. Additionally, the paths generated by the proposed algorithm also exhibit shorter path lengths compared to EGO-Swarm.

CONCLUSION

In this study, we introduce a novel hierarchical deep reinforcement learning algorithm aimed at dynamically adjusting the maximum speed and maximum acceleration of EGO-Swarm, an autonomous drone flight algorithm that has gained significant attention recently. The effectiveness of our proposed algorithm is evaluated through comparative experiments conducted in ROS simulation. Future research endeavors will focus on enhancing the real-time autonomous flight performance, particularly in scenarios involving swarm drones. Utilizing deep reinforcement learning enables adaptive optimization of parameter control in navigation algorithms like EGO-Swarm. Experimental findings demonstrate notable enhancements in drone navigation performance, resulting in faster, smoother, and more efficient routes with reduced path length. Consequently, this approach holds promising potential for enhancing drone applications across diverse fields.

REFERENCE

Jurnal Minfo Polgan
Volume 13, Nomor 1, 2024 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v13i1.13489 p-ISSN : 2089-9424

Alyassi, R., Khonji, M., Karapetyan, A., Chau, S. C.-K., Elbassioni, K., & Tseng, C.-M. (2022). Autonomous recharging and flight mission planning for battery-operated autonomous drones. *IEEE Transactions on Automation Science and Engineering*, *20*(2), 1034–1046.

de Jesus, J. C., Kich, V. A., Kolling, A. H., Grando, R. B., Cuadros, M. A. de S. L., & Gamarra, D. F. T. (2021). Soft actor-critic for navigation of mobile robots. *Journal of Intelligent \& Robotic Systems*, *102*(2), 31.

Duan, J., Guan, Y., Li, S. E., Ren, Y., Sun, Q., & Cheng, B. (2021). Distributional soft actor-critic: Off-policy reinforcement learning for addressing value estimation errors. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(11), 6584–6598.

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International Conference on Machine Learning*, 1861–1870.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., & others. (2018). Soft actor-critic algorithms and applications. *ArXiv Preprint ArXiv:1812.05905*.

Hafiz, S., Ginting, N., & Nababan, E. B. (n.d.). Performance improvement of ant colony optimization algorithm using multi-attribute rating simple technique exploiting ranks. *International Journal of Research and Review*, *7*, 150–154.

Nguyen, D. D., Rohacs, J., & Rohacs, D. (2021). Autonomous flight trajectory control system for drones in smart city traffic management. *ISPRS International Journal of Geo-Information*, *10*(5), 338.

Park, J., Kim, D., Kim, G. C., Oh, D., & Kim, H. J. (2022). Online distributed trajectory planning for quadrotor swarm with feasibility guarantee using linear safe corridor. *IEEE Robotics and Automation Letters*, *7*(2), 4869–4876.

Wayahdi, M. R., Ginting, S. H. N., & Syahputra, D. (2021). Greedy, A-Star, and Dijkstra's Algorithms in Finding Shortest Path. *International Journal of Advances in Data and Information Systems*, *2*(1), 45–52. https://doi.org/10.25008/ijadis.v2i1.1206

Yang, J., Liu, Y., Man, Y., Li, F., & Wang, Y. (2023). A PSO-Based Cooperative Search Method for Multi-UAVs with Collision Avoidance. *2023 42nd Chinese Control Conference (CCC)*, 5992–5997.

Zhou, X., Zhu, J., Zhou, H., Xu, C., & Gao, F. (2021). Ego-swarm: A fully autonomous and decentralized quadrotor swarm system in cluttered environments. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 4101–4107.