UAV Path Planning Based on Point Cloud Deep Reinforcement Learning

Yi Yang [1], Zheng Xie [2], Junhao Huang [3], Hanlei Li [4], Jiarun Gan [5]

[1]36920241153271 class for Institute of AI
[2]36920241153266 class for Institute of AI
[3]36920241153214 class for Institute of AI
[4]36920241153227 class for Institute of AI
[5]36920241153203 class for Institute of AI

Abstract

This project focuses on developing an autonomous driving algorithm for unmanned aerial vehicles (UAVs) based on Deep Reinforcement Learning (DRL), aimed at addressing the challenges of autonomous navigation and decision-making in complex environments. Traditional UAVs rely on predefined paths, which limits their effectiveness in dynamic and unknown situations. By incorporating deep reinforcement learning, our research will enable UAVs to learn in real-time and optimize their flight strategies, effectively navigating obstacles, weather changes, and dynamic targets. At the core of this study is a novel path planning algorithm, PPO-PointNet, based on deep reinforcement learning. This approach leverages a pre-trained PointNet model to extract three-dimensional features from point cloud data and integrates them into the Proximal Policy Optimization (PPO) framework. This enhancement improves the agent's perception capabilities, allowing the UAV to autonomously learn and adapt. Through interaction with the environment, the UAV will gradually discover optimal strategies, thereby enhancing its flexibility and intelligence in real-time decision-making. Through the implementation of this project, we aim to provide innovative insights into UAV autonomous flight technology, promoting its intelligent development in various complex tasks.

Introduction

Drone piloting refers to the process of controlling unmanned aerial vehicles (UAVs) during flight, with the goal of accomplishing specific tasks in various environments, such as aerial photography, package delivery, and surveillance.(Shakhatreh et al. 2019; Elmeseiry, Alshaer, and Ismail 2021) One of the key technologies involved is drone path planning, which aims to design an optimal flight route from the starting point to the destination(Zhao et al. 2020), ensuring the shortest time, minimal energy consumption, or maximum safety. Path planning must take into account obstacles that the drone may encounter during flight, such as buildings, trees, or other drones(Cai et al. 2020). Specifically, the integration of drone piloting and path planning is reflected in several key aspects:

- Autonomous Flight Control: The drone generates the optimal path from start to finish using path planning algorithms that consider factors such as flight time, energy consumption, and obstacle locations.(Kwak and Sung 2018) The flight control system then adjusts the drone's attitude and speed in real-time to ensure it follows the planned route stably, even in the face of changing external conditions like wind speed and airflow.(Gugan and Haque 2023)
- Dynamic Obstacle Avoidance: The system can continuously gather real-time environmental data—such as obstacle positions and weather conditions—using sensors like LIDAR and cameras(Liu et al. 2017). Based on this data, path planning algorithms can dynamically adjust the flight path to avoid potential collisions and hazards, enhancing flight safety. The drone can quickly respond and change its course to navigate around unpredictable obstacles.
- Feedback Mechanism: As the drone flies, it constantly monitors environmental changes and relays information back to the path planning system. (Zhang, Zhang, and Low 2021)This feedback allows for real-time optimization to address sudden environmental changes and unexpected situations, ensuring safety and efficiency in flight operations. (Xiang et al. 2023)For example, if a new obstacle is detected, the system can immediately recalculate the path and guide the drone to adjust its route safely.
- Integration of Intelligent Algorithms: The integration of intelligent algorithms, such as reinforcement learning, can further enhance the intelligence of path planning. This allows the drone to adaptively navigate complex environments, better responding to changes and challenges.(Azar et al. 2021)By analyzing historical data, the system can learn and optimize future path selections, improving overall flight efficiency and safety(Nishitani et al. 2015).

Through this combination, drones can efficiently follow predetermined paths while also responding in real-time to various challenges, enabling a wider range of applications, such as logistics delivery, environmental monitoring, agricultural spraying, and search and rescue operations. This intelligent development will empower drones to play increasingly important roles in diverse scenarios, enhancing operational efficiency, reducing labor costs, and driving progress and innovation in the industry.

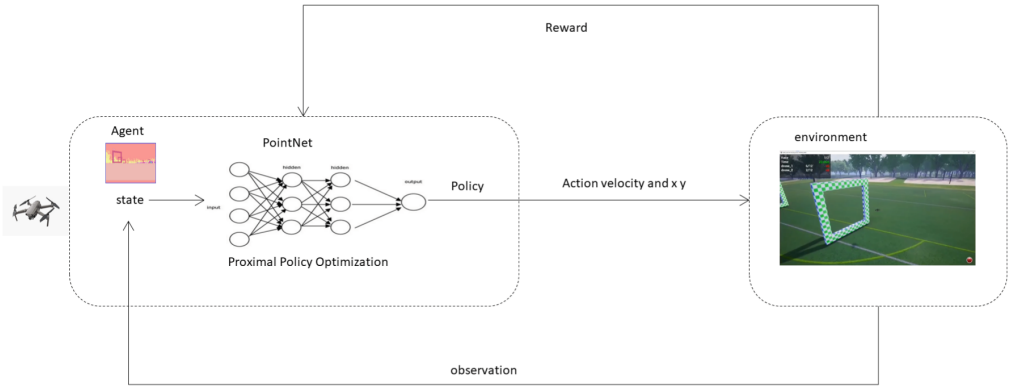


Figure 1: network structure

Related Work

Based on the environmental information obtained by robots during movement, path planning can generally be divided into two stages: global planning and local planning(Wang et al. 2017). Kuanqi Cai et al. proposed a navigation framework specifically designed for robots, which includes both a global path planner and a local path planner. The global path planner is primarily responsible for planning a collision-free path from the starting point to the target point. This process only involves static global mapping, resulting in static global paths that do not consider dynamic situations. In contrast, the local path planner incorporates dynamic information to optimize the segmentation of the global path.

There is substantial research on global path planning, with classical algorithms based on graph search, including Dijkstra's algorithm, A* algorithm, Depth First Search (DFS), and Breadth First Search (BFS). Over the past few decades, Dijkstra's and A* algorithms have been extensively studied and have been widely applied in real-world robotic applications through ROS (Robot Operating System)(**?**), demonstrating their effectiveness. These heuristic search strategies perform well in relatively simple two-dimensional environments, but they impose a significant computational burden when implemented in large-scale or high-dimensional environments.

To enhance computational efficiency and avoid local optima, researchers have proposed intelligent algorithms based on biomimicry to simulate the evolutionary behaviors of insects. These generally include Genetic Algorithms (GA), Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO). For instance, Wang et al. introduced a Genetic Algorithm-Particle Swarm Optimization (OGA-PSO) method to tackle the challenge of finding the shortest collision-free path for robots during path planning(Wang et al. 2016). Liu et al. integrated artificial potential fields and geometric local optimization techniques with the Ant Colony Algorithm to explore globally optimal paths(Liu et al. 2017).

The focus of local path planning is to utilize the environmental information around the robot to generate a local path. Given the real-time variations in sensor data within a dynamic environment, local path planning is extensively utilized. Nishitanti et al. proposed an X-Y-T space motion planning approach to evade obstacles, taking into account their orientation and personal domains(Nishitani et al. 2015). However, the computational efficiency is significantly influenced by the grid size of the navigation map. Additionally, as an enhancement of the A* algorithm, the Timed A* method anticipates the trajectories of human obstacles through the use of a social cost function(van Hasselt, Guez, and Silver 2015). Although local path planning algorithms have higher efficiency and practicality, a notable disadvantage is that local planners may get trapped in local minima.

To achieve optimal paths and avoid the local minimum issue, various approaches such as the artificial potential field method(Le Gouguec et al. 2017), fuzzy logic algorithms, simulated annealing algorithms, and particle algorithms have been proposed. However, these techniques often overlook the relative motion between agents and dynamic objects, making it challenging to determine the motion trajectories of dynamic objects in certain situations.

With the development and popularization of deep learning (DL) and reinforcement learning (RL), they have shown better performance in addressing complex nonlinear problems. The complexity of these problems typically refers to uncertainty, ambiguity, and incompleteness(Cai et al. 2017). Google DeepMind's introduction of the Deep Q-Network (DQN) represents a breakthrough(Mnih et al. 2013), utilizing a replay buffer to reuse old data and improve efficiency. However, the influence of noise on the estimation of state-action values (Q) limits its robustness. Consequently, Double DQN and Dueling DQN were developed to address issues caused by noise. Double DQN employs another network to evaluate the Q value estimates in DQN to reduce noise, while Dueling DQN uses the advantage value (A value) to obtain better Q values, significantly reducing noise. However, the implementation of these algorithms can be computationally expensive.

As a result, gradient methods are directly used to optimize the strategies for generating optimal behavior. The policy gradient method is relatively stable in terms of network convergence but lacks efficiency in convergence speed. The actor-critic architecture improves convergence speed, but this improvement often requires sacrificing convergence

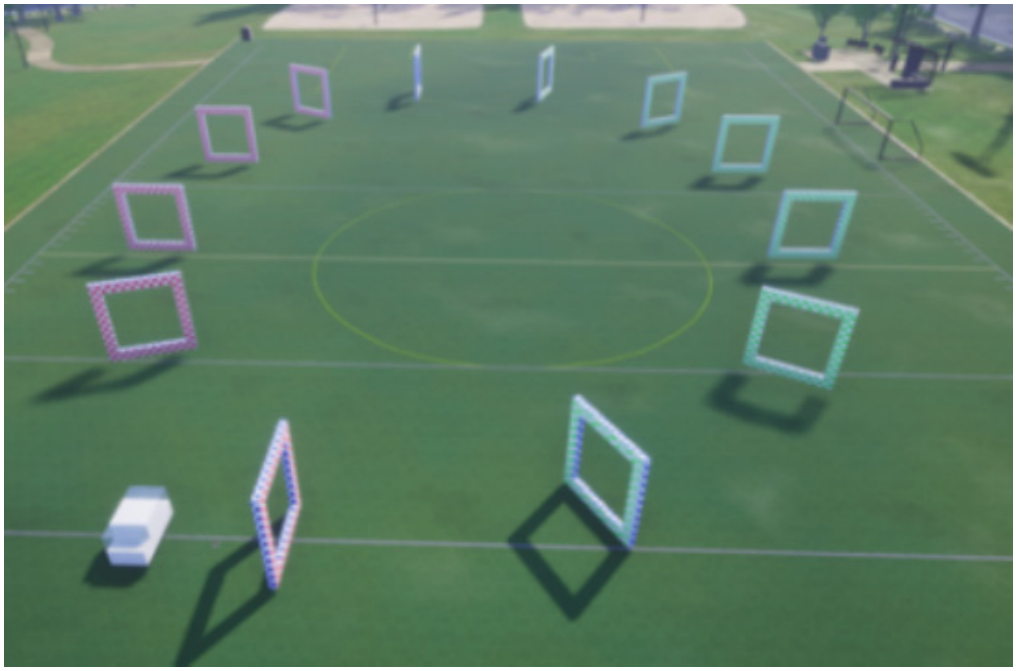


Figure 2: Simulation scene

stability, making it difficult for the network to converge in early training. Subsequently, Trust Region Policy Optimization (TRPO)(Schulman et al. 2015) and Proximal Policy Optimization (PPO)(Schulman et al. 2017) were introduced to address these shortcomings. In PPO, the concepts of "advantage" and adaptive penalties are introduced to enhance convergence speed and stability(Mnih et al. 2013). PPO is a well-known continuous control algorithm that outperforms traditional path planning algorithms when dealing with complex nonlinear problems. Furthermore, PPO has the advantages of easy implementation and fewer reusable hyperparameters. As an advantage-based actor-critic algorithm, PPO aims to maintain conservativeness in policy updates.

This paper intends to train the agent based on the PPO algorithm. By utilizing Kullback-Leibler divergence and a truncated surrogate function, multiple update steps in a trajectory can be performed without concerns about policy changes, making it particularly suitable for the AirSim simulation environment.

Recent advancements in 3D data acquisition have sparked significant interest in understanding point clouds.

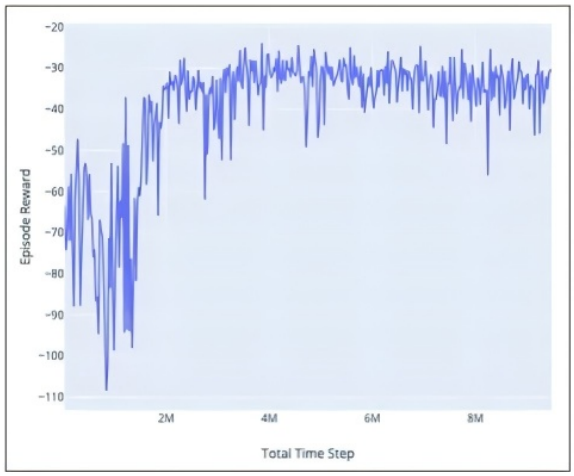
With the emergence of PointNet(Charles et al. 2017) and PointNet++(Qi et al. 2017), it has become feasible to use deep convolutional neural networks (CNNs) to process unstructured point cloud data. Following the introduction of "PointNets," numerous point-based networks have been developed, many of which focus on creating new complex modules for extracting local structures, such as the pseudo-grid convolution in KPConv(Thomas et al. 2019) and the self-attention layers in Point Transformer(Zhao et al. 2020). These novel methods have significantly outperformed PointNet++ across various tasks, suggesting that the PointNet++ architecture may be too simplistic to effectively learn complex representations of point clouds.

However, point clouds possess numerous advantages in 3D data acquisition that 2D data cannot match. For instance, point clouds consist of a collection of 3D coordinate points that can accurately represent the surfaces and geometric shapes of objects. This high-fidelity 3D representation is particularly well-suited for capturing complex and irregular shapes. Moreover, point cloud data can be integrated with RGB images, depth sensors, and other sensor data to form a more comprehensive environmental model, providing rich feature information. Leveraging these strengths, point clouds demonstrate unique value in applications such as robotics and autonomous driving.

Proposed Solution

Proximal policy optimization is a model-free off-policy behavior evaluation algorithm introduced by OpenAI. It utilizes the Clipped Surrogate function as the optimization objective. PPO is characterized by easy implementation and can achieve results comparable to other policy algorithms by adjusting fewer parameters. The core of PPO is to employ a method called Importance Sampling, which transforms the on-policy training process in policy gradient into off-policy training, that is, from online learning to offline learning. In experiments, the training speed and effect are significantly improved compared with the policy gradient, thus becoming one of the commonly used algorithms for continuous control problems. It is formally described as follows:

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right] = \mathbb{E}_t \left[r_t(\theta) \hat{A}_t \right]$$



(a) The work done by Ref

(b) The work done by us PPO-PointNet

Figure 3: Scene Rewards - Comparison of Time Steps during Training

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[min \left(r_t(\theta)\hat{A}_t, clip\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t \right) \right]$$

Here, $\hat{A}_t$ represents the generalized advantage function. According to whether the advantage is positive or negative, the hyperparameter ϵ is used to adjust the clipping ratio between $1-\epsilon$ and $1+\epsilon$. It is generally believed that the key innovation of PPO superior to TRPO lies in its ratio clipping mechanism being superior to the KL divergence.

Point clouds are an essential type of geometric data structure. Due to their irregular format, most researchers believe that such data must be converted into regular 3D voxel grids or image sets. However, this approach can result in large data sizes and associated complications. To address this issue, we first employ a point cloud segmentation network to separate the targets of interest from the irrelevant background. We then input the segmented 3D point cloud data into the PointNet neural network, which serves as both the Actor and Critic network within our PPO framework. This method allows for direct processing of point clouds while fully respecting the order invariance of the input points.

The PPO-based reinforcement learning model is implemented through an agent that obtains observations from the AirSim Drone Racing Lab simulation environment, specifically the 3D point cloud state generated by the LiDAR sensor mounted on the drone. We designed a neural network architecture called PointNet to extract 3D features from the point clouds. This network extracts these features to train the policy, enabling the drone to perform actions in the environment, such as flying through race gates and receiving rewards.

Meanwhile, point cloud data, as an essential part of the environmental model, is crucial for UAV decision-making. However, the original point cloud data is often affected by noise. To improve the accuracy and effectiveness of path planning, an advanced point cloud denoising method is introduced.

This method constructs a new paradigm based on the distribution characteristics of noisy point clouds. A noise-free point cloud can be regarded as a collection of samples from a 3D distribution supported by 2D manifolds. When affected by noise, its distribution becomes a convolution form of the original distribution and the noise model. Under certain assumptions of the noise model, the mode of this convolution result corresponds to the underlying clean surface and has a higher probability than the surrounding space. Therefore, the denoising process can be achieved by performing a gradient ascent operation on the relevant log-probability function to move the noisy points towards the mode.

To implement this denoising method, a specialized neural network architecture is designed to estimate the score of the input noisy point cloud distribution, that is, the gradient of the log-probability function. This network consists of a feature extraction unit and a score estimation unit. The feature extraction unit is composed of a series of densely connected dynamic graph convolutional layers, which can extract multi-scale, local and non-local features for each point, and the dense connection can produce features with rich contextual information. The score estimation unit is parameterized according to the point's features and takes the 3D coordinates near the point as input and outputs a score.

In the network training process, a corresponding objective function is set. Given the input noisy point cloud and the corresponding noise-free point cloud, a method for calculating the score of a point in 3D space is defined. The train.

Reward design is a significant area of research in reinforcement learning. Manual reward engineering is a common technique in developing such systems.

Our agent needs to pass through gates without colliding with them, so we retrieve the gate locations in the environment through API calls and sort them based on their relative

positions. We calculate the L2 distance between the midpoint of each gate and the drone, providing rewards based on the L2 distance to the next gate the drone needs to pass. For the distance between the drone and the gate frame, we designed a specialized potential field reward function. Additionally, we established a maximum distance threshold; if the distance exceeds this threshold, the game ends. A timer tracks the duration of each game, with a limit set at 300 seconds.

In the early stages of training, we observed that the agent sometimes collided with obstacles and became stuck, leading to poor data quality for subsequent training and significantly prolonging convergence time. Therefore, relying solely on the timer to determine game termination proved insufficient. To address this, we recorded both the elapsed time and the number of collisions, resetting the training environment whenever a collision was detected.

By using PointNet (the structure is shown as 1) to extract 3D features and train the PPO policy, we adopt a faster and more robust approach that effectively minimizes the impact of minor perturbations to input points, as well as issues related to point insertion or deletion.

Experiments

The AirSim Drone Racing Lab is a simulation framework for the rapid prototyping of autonomous algorithms and supports the use of machine learning for research. Its purpose is to reduce the risk of drones flying in the real world. This framework can generate racetracks in multiple realistic environments, arrange drone races, attach some doorframe obstacles, and allow multiple sensor modes and different camera models. In this paper, a rival drone equipped with the baseline algorithm provided by the framework itself is used in a simulated environment, which helps us in the training and evaluation process. At the same time, it is also easy to import 3D models, such as drones, door frames, and randomization domain obstacles. During the training process, we can obtain most of the drone information through the API, which allows us to obtain the current score, progress, and time penalty. The current environmental state and 3D point cloud are input into the network training strategy, which is used to control the drone to pass through the gate.

Our environment is selected as an open football field with a total of twelve square door frames. As shown in Figure 2 The drone should start from the starting point, go through all the door frames counterclockwise, and complete as quickly as possible without colliding with the opponent drone and the door frame.

We set the maximum step size in each episode to 30, calculate the total reward value to update the policy after 2 * maximum step size each time, and update the policy for 40 epochs in one PPO update. We set the starting std for action distribution (Multivariate Normal) to 0.6 and decay the std linearly to 0.05. The minimum std is set to 0.1, which is used to limit the scope of action exploration. The clip parameter for PPO is set to 0.2, and a higher discount factor for continuous actions $\gamma = 0.99$ is retained. Finally, the learning rates of the actor network and the critic network are both set to 10^{-4}.

Our goal is to hope that the agent can pass through the checkpoint of 12 gates within a given time limit and complete it as quickly as possible while avoiding collisions to the greatest extent. As shown in Figure 3, We found that the agent began to approach the performance of a real human operator after sufficient training time. After 15,000 steps in the simulation, the agent began to partially complete 12 checkpoints. According to our optimized reward engineering, our agent reached the level of work done by U. Ates (Ates 2020) and others, and the training time required was greatly reduced.

Conclusion

Unmanned aerial vehicles (UAVs) are becoming increasingly significant. However, existing algorithms frequently suffer from drawbacks such as lack of flexibility, high computational burden, and susceptibility to falling into local minimum traps. This paper proposes a novel algorithm that combines PPO and PointNet. By using the Clipped Surrogate mechanism to limit the magnitude of the policy gradient update and integrating our manually set reward project, the time spent on agent training is significantly reduced.

Experiments demonstrate that only approximately 15,000 rounds of training are required for the agent to achieve a level close to that of a real human operator. Future work will focus on exploring how to enable agents to reach a good performance level with a small amount of training in more complex environments.

References

Ates, U. 2020. Long-term planning with deep reinforcement learning on autonomous drones. In *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, 1–6. IEEE.

Azar, A. T.; Koubaa, A.; Ali Mohamed, N.; Ibrahim, H. A.; Ibrahim, Z. F.; Kazim, M.; Ammar, A.; Benjdira, B.; Khamis, A. M.; Hameed, I. A.; et al. 2021. Drone deep reinforcement learning: A review. *Electronics*, 10(9): 999.

Cai, K.; Wang, C.; Cheng, J.; De Silva, C. W.; and Meng, M. Q. H. 2020. Mobile Robot Path Planning in Dynamic Environments: A Survey. *arXiv e-prints*, arXiv:2006.14195.

Cai, M.; Lin, Y.; Han, B.; Liu, C.; and Zhang, W. 2017. On a Simple and Efficient Approach to Probability Distribution Function Aggregation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(9): 2444–2453.

Charles, R. Q.; Su, H.; Kaichun, M.; and Guibas, L. J. 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 77–85.

Elmeseiry, N.; Alshaer, N.; and Ismail, T. 2021. A detailed survey and future directions of unmanned aerial vehicles (uavs) with potential applications. *Aerospace*, 8(12): 363.

Gugan, G.; and Haque, A. 2023. Path planning for autonomous drones: Challenges and future directions. *Drones*, 7(3): 169.

Kwak, J.; and Sung, Y. 2018. Autonomous UAV Flight Control for GPS-Based Navigation. *IEEE Access*, 6: 37947–37955.

Le Gouguec, A.; Kemeny, A.; Berthoz, A.; and Merienne, F. 2017. Artificial Potential Field Simulation Framework for Semi-Autonomous Car Conception. In Kemeny, A.; Colombet, F.; Merienne, F.; and Espié, S., eds., *Proceedings of the Driving Simulation Conference 2017 Europe VR*, 19–22. Stuttgart, Germany: Driving Simulation Association.

Liu, J.; Yang, J.; Liu, H.; Tian, X.; and Gao, M. 2017. An improved ant colony algorithm for robot path planning. *Soft Computing*, 21(19): 5829–5839.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing Atari with Deep Reinforcement Learning. *arXiv e-prints*, arXiv:1312.5602.

Nishitani, I.; Matsumura, T.; Ozawa, M.; Yorozu, A.; and Takahashi, M. 2015. Human-centered X–Y–T space path planning for mobile robot in dynamic environments. *Robotics and Autonomous Systems*, 66: 18–26.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv e-prints*, arXiv:1706.02413.

Schulman, J.; Levine, S.; Moritz, P.; Jordan, M. I.; and Abbeel, P. 2015. Trust Region Policy Optimization. *arXiv e-prints*, arXiv:1502.05477.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *arXiv e-prints*, arXiv:1707.06347.

Shakhatreh, H.; Sawalmeh, A. H.; Al-Fuqaha, A.; Dou, Z.; Almaita, E.; Khalil, I.; Othman, N. S.; Khreishah, A.; and Guizani, M. 2019. Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. *IEEE Access*, 7: 48572–48634.

Thomas, H.; Qi, C. R.; Deschaud, J.-E.; Marcotegui, B.; Goulette, F.; and Guibas, L. 2019. KPConv: Flexible and Deformable Convolution for Point Clouds. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 6410–6419.

van Hasselt, H.; Guez, A.; and Silver, D. 2015. Deep Reinforcement Learning with Double Q-learning. *arXiv e-prints*, arXiv:1509.06461.

Wang, C.; Meng, L.; She, S.; Mitchell, I. M.; Li, T.; Tung, F.; Wan, W.; Meng, M. Q.-H.; and de Silva, C. W. 2017. Autonomous mobile robot navigation in uneven and unstructured indoor environments. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

Wang, X.; Shi, Y.; Ding, D.; and Gu, X. 2016. Double global optimum genetic algorithm–particle swarm optimization-based welding robot path planning. *Engineering Optimization*, 48: 299 – 316.

Xiang, H.; Han, Y.; Pan, N.; Zhang, M.; and Wang, Z. 2023. Study on multi-UAV cooperative Path planning for complex patrol tasks in large cities. *Drones*, 7(6): 367.

Zhang, N.; Zhang, M.; and Low, K. H. 2021. 3D path planning and real-time collision resolution of multirotor drone operations in complex urban low-altitude airspace. *Transportation Research Part C: Emerging Technologies*, 129: 103123.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P.; and Koltun, V. 2020. Point Transformer. *arXiv e-prints*, arXiv:2012.09164.