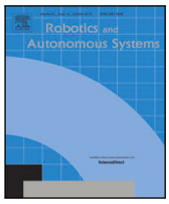
Robotics and Autonomous Systems 207 (2027) 105833



Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot



Mapless UAV navigation in complex environments: A survey on deep reinforcement learning approaches

Dawei Ni [a,b], Steven H.H. Ding [a], Benjamin C.M. Fung [a,*]

[a] School of Information Studies, McGill University, Canada
[b] AI Acceleration Hub, Ericsson, Canada

ARTICLE INFO

Keywords:
Unmanned Aerial Vehicles
AI
Mapless
Deep learning

ABSTRACT

As applications of Unmanned Aerial Vehicles (UAVs) continue to expand across sectors such as logistics, surveillance, and environmental monitoring, the development of safe and robust autonomous navigation capabilities has become increasingly important. Mapless autonomous UAV navigation refers to a UAV's ability to reach a designated destination along a collision-free and efficient path without human intervention and without relying on a global map. Achieving this capability is particularly challenging in complex environments, especially in cluttered spaces, in the presence of dynamic obstacles, or under wind disturbances. Deep reinforcement learning (DRL) has emerged as one of the leading approaches for modeling UAVs as intelligent agents that learn effective control policies through interaction with the environment. However, the performance of these methods depends strongly on both the characteristics of the environment and the limitations of the underlying algorithms. A deep understanding of algorithmic trade-offs is essential for selecting and tailoring the appropriate RL framework to ensure safe, robust and efficient UAV navigation in complex environments. Within this context, this study provides a structured overview of recent advances in state-of-the-art DRL approaches for Mapless UAV navigation and reviews their application across various UAV-environment dynamics and constraints. In addition, the review identifies key challenges in deploying DRL in highly complex environments. It also outlines open research questions and proposes future research directions to advance this field.

1. Introduction

Unmanned Aerial Vehicles (UAVs) have experienced rapid growth in recent years, driven by their operational flexibility and growing applications across diverse domains, including package delivery, traffic monitoring, environmental surveillance, and search and rescue [1]. This rapid growth in UAV applications is driven by inherent benefits such as reduced operational costs, minimized human risk, and improved efficiency, making them indispensable for modern robotics and infrastructure development [1,2].

Autonomous UAV navigation refers to a UAV's ability to carry out tasks, such as flying between two points by using onboard sensors or cameras to navigate, avoid obstacles, and determine paths without human intervention [3,4]. This form of autonomy is part of the broader category of Autonomous Systems (AS), which includes self-driving cars, drones, and robots, all designed to operate efficiently while reducing risks linked to human intervention [5,6]. Equipped with advanced control algorithms and computing platforms, autonomous UAVs are capable of making precise, real-time decisions in complex environments that were once difficult to navigate. In practical applications, UAVs must operate safely in these environments to complete assigned tasks. As a result, ensuring safe and efficient navigation has become a central focus of research across various operational scenarios. By reducing the need for constant human oversight, these systems enhance operational speed and reliability while opening new opportunities for innovation, ultimately contributing to safer and more sustainable advancements across society [7].

UAVs face major challenges in achieving autonomous navigation, especially in unknown environments. Traditional non-learning-based methods, such as A* [8], Rapid-Exploring Random Tree (RRT) [9], Artificial Potential Field (APF) [10], Hybrid A* [11], and Genetic Algorithm [12], rely heavily on complete prior knowledge of the environment to plan feasible paths. Similarly, classic control techniques like PID-based control [13], Sliding Mode Control (SMC) [14], Backstepping [15] and RISE [16] require precise system models and focus primarily on short-term error minimization. These methods often fall short in real-world scenarios due to environmental uncertainties, sensor noise, imperfect actuation, state estimation errors, unmodeled dynamics, and external disturbances. They also struggle to achieve optimal long-term performance in complex UAV missions.

* Corresponding author.
E-mail addresses: dawei.ni@mail.mcgill.ca (D. Ni), steven.h.ding@mcgill.ca (S.H.H. Ding), ben.fung@mcgill.ca (B.C.M. Fung).

https://doi.org/10.1016/j.robot.2026.105833
Received 16 April 2026; Received in revised form 2 October 2026; Accepted 25 October 2026
Available online 31 October 2026
0921-8890/© 2026 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

Complex environments present more serious challenges to UAV reliability and real-world deployment. In high-density areas like dense forests or pedestrian zones, UAVs must detect and react to moving obstacles in real time. These dynamic elements produce large volumes of interactive data, making the training of end-to-end neural networks considerably more difficult [17,18]. For instance, frequent collisions with dynamic obstacles during early training stages lead to many failed episodes and negative rewards, and consequently limit positive experiences for learning effective navigation policies. Additionally, wind disturbances add further complexity. These stochastic and dynamic forces can cause significant deviations from intended flight paths, reducing control accuracy, increasing energy consumption, and compromising mission success [16,19]. Supervised learning (SL) approaches use deep learning to map sensor or camera inputs directly to navigation commands [20–23]. These approaches rely on onboard devices to gather environmental data and determine appropriate control actions. While SL methods perform well in controlled environments, their dependence on labeled datasets and limited adaptability to changing conditions reduce their effectiveness and robustness in dynamic settings.

Deep Reinforcement Learning (DRL) [24], which combines Reinforcement Learning (RL) [25] with deep learning, has emerged as a leading approach to address these challenges. RL agents are not explicitly instructed regarding correct actions. Instead, they must explore their environment through trial and error to learn policies that maximize long-term rewards [25]. This exploration-based approach makes RL particularly well-suited to tasks characterized by uncertainty or incomplete prior knowledge. Unlike traditional methods, DRL is capable of end-to-end learning by mapping raw sensor observations directly to actions, bypassing the need for accurate robot or sensor models and mitigating uncertainties like sensor noise [26,27]. Typically, model-free DRL approaches are used due to their ability to learn optimal policies without requiring prior knowledge of the environment. These algorithms adapt effectively to environmental changes through trial-and-error interactions [19,28]. Furthermore, many DRL methods incorporate historical state observations to capture spatial–temporal patterns, reducing oversimplification and enabling agents to more effectively and reliably predict and adapt to moving obstacles in complex environments [19,29,30]. Compared to SL, DRL eliminates the need for time-consuming labeled datasets, learning directly from environmental feedback, which enhances its flexibility and robustness.

1.1. Scope and contribution

This survey aims to highlight major developments from the past 5–10 years, with a primary focus on contributions from the last five years, while also incorporating key earlier approaches for completeness. The research community has published numerous reviews and surveys examining different challenges in UAV navigation. For example, several comprehensive studies have investigated broader UAV applications and system implementations [1,31]. Other surveys have focused on the communication aspects of UAV wireless networks [2,32]. RL or DRL approaches have been the sole focus of dedicated reviews [28,33,34]. Surveys [3,35] on AI-based approaches in UAV applications offer an overview of a wide range of machine learning techniques, including methods for perception, planning, and control. UAV control methodologies are illustrated and discussed in [36,37]. Path-planning algorithms specific to navigation have been discussed in [38,39]. Surveys on general autonomous UAV navigation [7] present holistic perspectives on UAV autonomous navigation systems. Additionally, studies focused on vision-based navigation [4,26,40] explore camera-based approaches for obstacle detection and localization in environments where GPS is unavailable. Table 1 provides a comparative analysis of recent related surveys including their research topic and main focus.

Despite the extensive body of literature, important gaps remain. To the best of the authors' knowledge, no dedicated survey exists focusing specifically on the application of RL to UAV navigation in complex environments. These environments pose unique real-time challenges, including cluttered spaces, moving obstacles, and wind disturbances. This paper addresses this gap by providing a comprehensive review of RL algorithms designed for autonomous UAV navigation under such conditions. To provide an overview of how existing studies address these challenges, Fig. 1 summarizes DRL design patterns among the studies reviewed in this work in terms of the composition of the base DRL algorithms and the major design techniques identified across the three environment categories. The contribution not only consolidates fragmented knowledge but also provides guidance for future research aimed at advancing UAV autonomy. More specifically, this work makes the following key contributions:

- The survey introduces a structured classification of complex UAV environments, providing insight into their specific operational challenges.
- The survey provides a comprehensive overview of major DRL approaches for mapless UAV navigation in complex environments, summarizing their core design choices and limitations.
- The survey outlines future research directions to effectively address the challenges identified within this study.

1.2. Research questions and search strategy

This survey adopts a structured narrative review approach to examine recent developments in DRL for mapless UAV navigation in complex environments. The objective is to identify representative and influential studies and synthesize the main methodological developments, design choices, and remaining challenges. The review is guided by the following research questions:

- What DRL algorithms and learning frameworks have been applied to mapless UAV navigation in complex environments?
- How are DRL methods adapted to different environmental challenges, particularly cluttered spaces, dynamic obstacles, and wind disturbances?
- What common techniques are used to improve perception, temporal awareness, exploration, learning efficiency, safety, and robustness in these environments?
- What limitations remain in current approaches, particularly regarding realistic control, generalization, sensing uncertainty, and real-world deployment?

The literature was identified through keyword-based searches of major scholarly databases and search engines, including IEEE Xplore, Scopus, ACM Digital Library, and Google Scholar, complemented by backward and forward citation searches of relevant studies and existing UAV navigation surveys. The search primarily focused on publications from the most recent five years, while earlier studies were retained when they introduced influential algorithms, problem formulations, or techniques that remain important to subsequent research.

Search terms were formed by combining UAV-related keywords, such as "UAV," "drone," "quadcopter," and "unmanned aerial vehicle," with learning and navigation terms including "deep reinforcement learning," "reinforcement learning," "autonomous navigation," "mapless navigation," "obstacle avoidance," and "local path planning." Environment-specific terms such as "cluttered environment," "dynamic obstacle," "moving obstacle," "wind disturbance," and "uncertain environment" were also used. Additional searches used algorithm-specific keywords, including DQN, DDPG, TD3, SAC, and PPO, to capture relevant variants and methodological developments.

Studies were retained when DRL formed a central component of UAV navigation, obstacle avoidance, motion planning, or disturbance-aware control in complex environments. Works primarily concerned with UAV communication, networking, task allocation, or purely global map-based planning were outside the main scope of this survey.

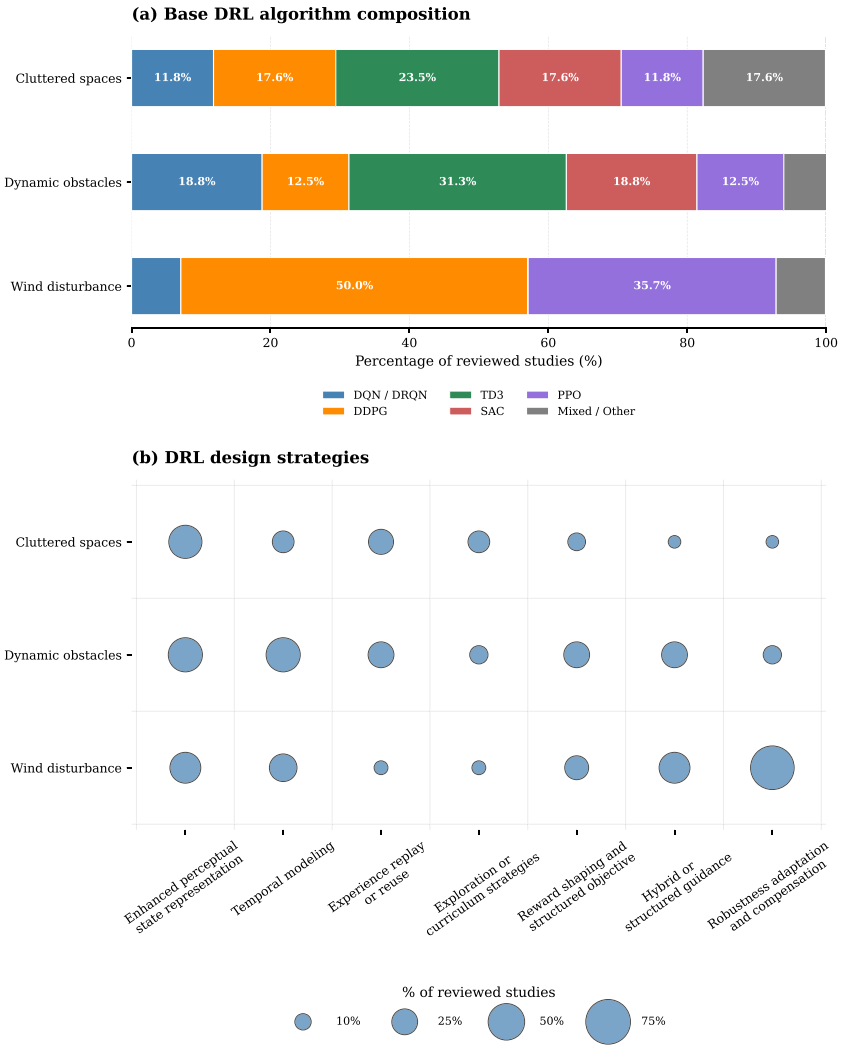


Fig. 1. DRL design patterns among the studies reviewed for different complex environments. (a) Composition of base DRL algorithms within the reviewed studies. (b) Relative occurrence of major design strategies identified in the reviewed literature. Percentages are calculated within each environment category and are intended to illustrate methodological patterns in the selected literature rather than field-wide prevalence.

1.3. Organization

This survey is organized as follows. Section 2 introduces the essential background, including UAV fundamentals, a brief overview of reinforcement learning (RL) and deep reinforcement learning (DRL), and the core components of autonomous UAV navigation. Section 3 presents a detailed discussion of DRL for autonomous UAV navigation in complex environments. It begins by describing UAV navigation environments and then reviews the various conditions in which UAVs operate, with particular attention to DRL-based adaptations for complex scenarios. Section 4 summarizes the simulation environments and performance metrics used to evaluate DRL methods. Section 5 discusses current research challenges and highlights promising directions for future work. Section 6 concludes the paper.

2. Background

This section provides a foundational overview of the background to contextualize DRL approaches for autonomous unmanned aerial vehicle (UAV) navigation in complex environments. The discussion is organized into three main subsections: (1) Fundamentals of UAV Types, Kinematics, and Dynamics (2) Reinforcement Learning and Deep Reinforcement Learning and (3) UAV Navigation Environment.

2.1. Fundamentals of UAV types, kinematics, and dynamics

2.1.1. UAV types

The classification of UAV is commonly based on several criteria, including control configuration, physical size, and levels of autonomy. These factors critically influence operational capabilities and the integration of advanced navigation techniques, such as DRL.

In terms of control configuration, UAVs fall into four primary categories [7]:

- Fixed-wing UAVs, known for their high endurance and speed, though limited by nonholonomic constraints that restrict maneuverability.
- Rotary-wing UAVs, including single-rotor types (e.g., helicopters) and multi-rotor platforms such as quadcopters, hexacopters, and octocopters, which provide vertical take-off and landing (VTOL) capabilities and agility in confined environments.
- Hybrid VTOL-fixed-wing UAVs, which combine the advantages of endurance and VTOL.
- Bio-inspired flapping-wing UAVs, designed for stealth and efficiency in constrained spaces.

Among these, quadcopters are the most widely used multi-rotor UAVs in civilian domains due to their structural simplicity, cost-effectiveness, and suitability for tasks such as aerial photography,

Table 1
Comparison of UAV review papers.

Refs.	Research topic	Year	Main focus
[31]	Recent UAV Advances	2022	General advances in UAV structure, navigation, and applications; recent developments in control systems and diverse uses such as search and rescue missions
[1]	Practical UAV Application	2023	UAV practical aspects, applications, challenges, security issues, and future trends in diverse fields
[32]	RL for Multi-UAV Networks	2023	RL-based approaches for autonomous multi-UAV wireless networks including effective operation in decision-making and multi-objective optimization under diverse conditions
[2]	UAV Communication and Control	2023	Communication and control advances in collaborative UAVs for enhanced coordination and performance
[28]	RL for UAV Navigation	2022	RL approaches for autonomous UAV navigation; classification of algorithms by environment and capabilities, and provides guidance on algorithm selection
[33]	Deep RL for UAV Application	2021	Deep reinforcement learning techniques for UAV guidance, navigation, and control; focus on autonomy in civilian and military applications; identifies current limitations and challenges for real-world deployment
[34]	RL for UAV Systems	2025	Comprehensive overview and summary of the application of RL in different UAV scenarios based on the classification of RL methods
[3]	AI for UAV Navigation	2022	AI methods for UAV navigation, addressing advances and future challenges in autonomy
[35]	ML for UAV Navigation	2024	Machine learning techniques for UAV navigation; focuses on real-time applications and improvements in energy efficiency, obstacle avoidance, communication, execution time, and path planning across different environments
[36]	UAV Control	2022	brief review of the UAV control issues, including motion equations, various classical and advanced control approaches
[37]	Quadrotor UAV Control	2023	Overview of Quadrotor UAV design including classical, intelligent, and hybrid controllers; covers modeling, applications, experimental comparisons
[38]	UAV Path Planning	2020	UAV path planning methods, solutions, and challenges for optimal collision-free paths
[39]	UAV Path Planning with Complexity Consideration	2023	Introduces environment complexity classification, current methods and highlights challenges in modeling, sensing, and collision avoidance
[7]	Autonomous UAVs navigation	2021	Recent advances in safe autonomous UAV navigation; focus on 3D obstacle avoidance and trajectory planning; highlights key challenges in achieving fully autonomous mission execution with minimal human input
[26]	Vision-Based Learning for UAV	2024	Vision-based learning methods with focus on operational capabilities across various scenarios; categorizes UAV control strategies ranging from single-agent systems to more complex multi-agent and heterogeneous system scenarios.
[4]	Vision-Based UAV Navigation	2018	Vision-based techniques for UAV localization, mapping, and obstacle avoidance; highlights advantages over laser/radar;
[40]	Vision-Based UAV Navigation	2023	Vision-based UAV navigation techniques covering visual localization, mapping, obstacle avoidance, and path planning; categorizes and compares methods; discusses challenges and open issues in GPS-denied environments.

delivery, and precision agriculture [41,42]. Hexacopters offer greater stability and redundancy, which make them ideal for mapping and search-and-rescue operations. In contrast, Octocopters are capable of carrying heavier payloads, are often used in infrastructure inspection and environmental monitoring. Overall, multi-rotor UAVs outperform fixed-wing and other configurations in civilian applications due to their user-friendly control, lower operational costs, and strong compatibility with autonomous navigation in dynamic environments [43].

Another common classification of UAVs is based on size, following standards set by aviation authorities such as the Australian Civil Aviation Safety Authority (CASA) and the U.S. Federal Aviation Administration (FAA). This classification includes five categories [44]:

- micro UAVs: less than 250 grams, typically used for indoor tasks
- very small UAVs: 0.25–2 kg
- small UAVs: 2–25 kg
- medium UAVs: 25–150 kg
- large UAVs: over 150 kg, typically used for military and long-endurance missions

In addition to weight, the physical size of UAVs, particularly frame diameter is closely related to their application and payload capacity. For example, quadcopters typically fall into the micro or small category. They often have frame diameters around 400 mm, which make them suitable for lightweight operations. By contrast, octocopters which are designed to carry heavier payloads often exceed 1000 mm in frame diameter to accommodate larger propellers and structural requirements [42]. These size-based distinctions are critical for regulatory compliance, operational planning, and integration with autonomy frameworks.

2.1.2. UAV kinematics

The kinematic modeling of UAVs involves a fundamental analysis of their translational and rotational motions within three-dimensional

space. Kinematics serves as the core theoretical framework describing UAV movements without explicitly considering forces and torques acting upon them. Fundamentally, UAV kinematics is typically described using a six degrees of freedom (6 DOF) model consisting of three translational motions (surge, sway, and heave) and three rotational motions (roll, pitch, and yaw) [45]. Translational kinematics characterizes linear displacement and velocity transformations in the inertial coordinate system, mathematically represented by the relationship:

$$\dot{\mathbf{p}} = \mathbf{R}\mathbf{v}, \tag{1}$$

where $\mathbf{p} = [x, y, z]^T$ denotes the position vector of the UAV in an inertial frame, $\mathbf{v} = [u, v, w]^T$ is the velocity vector expressed in the body-fixed frame, and $\mathbf{R}$ is a rotation matrix describing the UAV's orientation relative to the inertial frame. Typically this rotation matrix is parameterized by Euler angles roll (ϕ), pitch (θ), and yaw (ψ), and takes the following form of:

$$\mathbf{R} = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{2}$$

where c and s denote cosine and sine functions, respectively [46]. Rotational kinematics describes the angular motion and orientation attitude evolution of unmanned aerial vehicles (UAVs) through the equation:

$$\dot{\boldsymbol{\eta}} = \mathbf{T}_B \boldsymbol{\omega}, \tag{3}$$

where $\dot{\boldsymbol{\eta}}$ represents the time derivatives of the Euler angles (roll ϕ, pitch θ, yaw ψ), $\boldsymbol{\omega} = [p, q, r]^T$ is the angular velocity vector in the body-fixed frame, $\mathbf{T}_B$ is the transformation matrix converting body angular velocity to Euler angle rates [47]. The matrix $\mathbf{T}_B$ is given by:

$$\mathbf{T}_B = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \tag{4}$$

These relationships demonstrate how rotational motion influences UAV orientation, which is critical for trajectory planning and stabilization control. It should be noted that this Euler-angle-based formulation is subject to a kinematic singularity, thus in those case such as aggressive maneuvers quaternion-based representations are often preferred. In autonomous navigation, accurate kinematic models support essential functions such as trajectory generation, real time pose estimation, and precise maneuvering. For instance, quadrotors rely on kinematic equations to compute desired attitudes and positions, enabling stable hovering, agile movement, and effective obstacle avoidance. Fixed wing UAVs also use kinematic models for navigation, though they are subject to aerodynamic constraints that limit their maneuverability and minimum operational speeds [44]. As a result, UAV kinematics provides a fundamental mathematical framework that supports precise, dependable, and robust autonomous navigation.

2.1.3. UAV dynamics

UAV dynamics involve the analysis of forces, torques, and motions that govern the behavior of aerial vehicles, which form a crucial connection between kinematic modeling and practical autonomous control. While kinematics describes motion without considering its causes, dynamics incorporate mass, inertia, and external forces such as gravity, thrust, aerodynamic drag, and environmental disturbances (e.g. wind gusts) to model UAV behavior in three-dimensional space. For general rigid-body UAVs, the Newton Euler formalism describes the dynamics using six degrees of freedom (6 DOF). The translational dynamics can be expressed as follows:

$$m\dot{\mathbf{v}}_\mathbf{i} = \mathbf{R}\mathbf{f}_b + m\mathbf{g} + \mathbf{d}_d, \tag{5}$$

where m is the mass, $\mathbf{v}_\mathbf{i}$ is the inertial-frame velocity, $\mathbf{R}$ is the rotation matrix, $\mathbf{f}_b$ represents forces in the body frame, $\mathbf{g}$ is the gravitational acceleration, and $\mathbf{d}$ denotes external disturbances. The rotational dynamics are given by:

$$\mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} = \boldsymbol{\tau}_b + \boldsymbol{\tau}_d, \tag{6}$$

where $\mathbf{I}$ is the inertia matrix expressed in the body frame, $\boldsymbol{\omega}$ is the angular velocity, $\boldsymbol{\tau}_b$ is the control torque in the body frame, and $\boldsymbol{\tau}_d$ represents disturbance torques. These nonlinear and coupled equations highlight key challenges in UAV control, including underactuation and sensitivity to external perturbations, which are critical considerations for realistic simulation and autonomous operation in dynamic environments.

Multirotor UAVs, including quadcopters, hexacopters, and octocopters, are widely used in civilian applications due to their vertical takeoff and landing (VTOL) capabilities, mechanical simplicity, and high maneuverability [1]. These attributes make them particularly well-suited for autonomous navigation in complex urban or cluttered environments. The dynamics of multirotor UAVs are governed by the thrust and torques generated by individual rotors. For a quadcopter, each of the four rotors produces upward thrust $T_i = k_f\omega_i^2$ and reactive torque $Q_i = k_q\omega_i^2$, where ω_i is the rotor speed and k_f, k_q are motor-dependent constants [46]. Assuming a specific quadrotor geometry, rotor numbering, body-axis convention, and rotor spin-direction convention, the total thrust and body torques can be written as follows,

$$F = \sum T_i,$$

while the control torques about the body axes are computed as:

- Roll: $\tau_x = l(T_2 - T_4)$
- Pitch: $\tau_y = l(T_1 - T_3)$
- Yaw: $\tau_z = Q_1 - Q_2 + Q_3 - Q_4$

where l is the arm length from the center to each rotor. Due to the underactuated nature of the system (four control inputs for six degrees of freedom), translational and rotational dynamics are inherently coupled. For instance, forward motion requires tilting, introducing nonlinear behaviors such as aerodynamic drag, commonly modeled as $\mathbf{d} = -k_d\mathbf{v}_\mathbf{i}$, where k_d is a drag coefficient. Hexacopters and octocopters provide additional redundancy and fault tolerance, improving system reliability under dynamic conditions [48].

In autonomous UAV navigation, the dynamics of multirotor systems play a critical role. For instance, DRL policies often aim to learn mappings from system states such as position, velocity, and orientation to control actions (e.g. rotor speeds) by optimizing cumulative rewards. This enables capabilities such as stable hovering, agile maneuvering, and obstacle avoidance in non-stationary and unpredictable environments. Accurate dynamic models are essential for reducing the sim-to-real gap, thereby allowing DRL policies to generalize to real-world conditions that include disturbances such as wind gusts, sensor noise, or payload variations. These considerations are particularly crucial in mission-critical applications like search and rescue or autonomous package delivery. Many existing models make simplified assumptions such as the exclusion of blade flapping and aerodynamic interactions, which can affect performance and consistency [36]. Addressing this limitation continues to be an active area of research.

2.2. Reinforcement learning and deep reinforcement learning

2.2.1. Reinforcement learning

Reinforcement learning (RL) is a branch of machine learning where an agent learns to make decisions step by step by interacting with an environment. The goal is to maximize cumulative rewards, and the foundations of RL draw from behavioral psychology and optimal control theory [25]. Unlike supervised learning, which depends on labeled data, RL focuses on trial-and-error learning, where the agent improves its behavior based on rewards or penalties. This makes RL well-suited for dynamic applications such as game playing, robotics, and

autonomous systems. Most RL problems are described using Markov Decision Processes (MDPs), which assume that the full state of the system is observable. An MDP is defined by a tuple (S, A, P, R, γ), where S is the set of states, A is the set of actions, P is the transition probability, R is the reward function, and γ is the discount factor for future rewards [25]. When the agent cannot fully observe the state, the problem is better represented as a Partially Observable Markov Decision Process (POMDP). POMDPs extend MDPs by adding an observation space O and emission probabilities, which model situations where the agent only receives noisy or incomplete data [49]. This is common in real-world scenarios such as UAV navigation with limited or uncertain sensor inputs. Solving POMDPs is harder than solving MDPs, because the agent must estimate belief states to track likely true states. In practice, approximate methods are used, including spectral approaches or deep RL extensions [49].

In RL, two central ideas are value functions and policies. The value function, written as $V(s)$ for state values or $Q(s, a)$ for action values, estimates the expected long-term reward from a state or a state–action pair when following a policy. This allows the agent to judge the quality of its decisions using the Bellman equation:

$$V(s) = \mathbb{E}\left[R_{t+1} + \gamma V(s_{t+1}) \mid s_t = s\right] \tag{7}$$

A policy defines how the agent behaves by mapping states to actions. A policy can be deterministic (fixed action per state) or stochastic (probabilistic distribution over actions). Different methods use these concepts in different ways. Value-based methods, such as Q-learning aim to learn the optimal value function and then derive the policy from it. Policy-based methods directly optimize the policy parameters to maximize expected rewards. In many cases, both approaches are combined in actor–critic methods, where the actor updates the policy and the critic evaluates it. This combination often improves learning stability and efficiency [50].

One of the core challenges in RL is the balance between exploration and exploitation. Exploitation means selecting actions that are already known to produce high rewards based on current experience. Exploration means trying new actions to search for better outcomes, which is important for avoiding local optima [25]. Several strategies have been developed to manage this tradeoff. For instance, the ϵ-greedy method introduces randomness by choosing a random action with probability ϵ and the Upper Confidence Bound (UCB) method uses uncertainty estimates to guide decisions [51]. More advanced approaches, such as Uncertainty-based or Noise-based exploration allow the level of exploration to change dynamically across different tasks [52]. In deep RL, exploration is often supported through intrinsic rewards or curiosity-driven mechanisms [52], which encourage the agent to act even in environments where external rewards are sparse. These techniques help the agent discover useful behaviors that might not appear immediately rewarding.

2.2.2. Deep reinforcement learning

DRL extends classical RL by using deep neural networks as function approximators. This makes it possible for agents to manage high-dimensional and complex state spaces, which traditional approaches like tabular Q-learning cannot handle effectively due to the "curse of dimensionality" [25]. Deep RL builds on the main ideas of RL, including MDPs, POMDPs, value functions, policies, and the balance between exploration and exploitation. With neural networks, DRL can approximate value functions such as $Q(s, a)$ or directly learn policies, which allows scalable learning from raw sensory inputs like images or sensor data. This integration has transformed many fields. In robotics, gaming, and autonomous systems, DRL enables agents to learn end-to-end mappings from observations to actions through trial-and-error interaction with the environment. One of the most influential milestones was the introduction of the Deep Q-Network (DQN) in 2015 [53]. By combining Q-learning with convolutional neural networks, DQN reached human-level performance on Atari games and demonstrated the effectiveness of this approach.

One important element for addressing instability in Deep RL is experience replay. This mechanism stores the agent's interactions with the environment (state, action, reward, next state) in a replay buffer and samples mini-batches randomly for training [53]. By doing so, it breaks the temporal correlations in sequential data that can otherwise lead to biased gradients and inefficient online learning. It also improves sample efficiency by allowing past experiences to be reused multiple times. Experience replay helps reduce catastrophic forgetting and stabilizes neural network updates, which is especially useful in off-policy methods where the behavior policy differs from the target policy. A well-known variant is prioritized experience replay, which samples more often from transitions with higher temporal-difference errors, speeding up convergence.

Another essential element is the target network. This secondary neural network updates more slowly than the primary network through periodic or soft updates. It provides stable bootstrap targets for value estimation and reduces overestimation bias in Q-learning [54]. The delayed updates prevent the moving target problem, where fast parameter changes can destabilize training. In actor–critic frameworks, Deep RL uses deep networks for both the actor (policy) and the critic (value function). An example is the Deep Deterministic Policy Gradient (DDPG) algorithm, which is widely used for continuous control tasks [55]. Another notable off-policy, deterministic actor–critic method designed for continuous control is Twin Delayed Deep Deterministic Policy Gradient (TD3) [56], which extends DDPG to stablize learning. Other key techniques include the use of recurrent architectures like LSTMs for POMDPs to handle partial observability [57], and distributional RL, which models return distributions instead of expectations to improve robustness in uncertain environments [58].

DRL plays a pivotal role in enabling autonomous unmanned aerial vehicle (UAV) navigation in complex environments through algorithms like DQN or DDPG. DRL models UAV navigation as a MDP or POMDP, where the UAV learns to balance exploration of unknown terrains with exploitation of learned policies and adapt in real-time to avoid collisions and optimize energy efficiency. This approach outperforms traditional methods in handling uncertainties and partial observability in the navigation task. We will cover these applications including specific DRL variants and challenges in detail in later sections.

2.3. Core components in autonomous UAV navigation

2.3.1. Autonomy levels

According to the National Institute of Standards and Technology (NIST) [59], UAV autonomy levels range from manual operation to complete autonomy. At the lowest level, remotely controlled UAVs are manually piloted within line-of-sight without onboard sensing. Teleoperated UAVs allow for beyond-line-of-sight operation using sensor feedback but lack autonomous obstacle avoidance. Semi-autonomous systems operate under high-level human supervision, with intermittent autonomy. At the highest level, fully autonomous UAVs independently adapt to environmental changes without human intervention. Higher autonomy levels require increasingly sophisticated hardware and system integration. Fully autonomous UAVs depend on a range of advanced sensors, including LiDAR, cameras, inertial measurement units (IMUs), and GNSS integrated within attitude and heading reference systems (AHRS) or inertial navigation systems (INS) for accurate perception and navigation [7]. Flight controllers such as PIXHAWK support autopilot functionality while high-performance onboard processors enable real-time decision-making [60]. Additionally, energy-efficient components like Li-Po batteries and brushless DC motors are essential to sustain longer flights under dynamic and nonlinear conditions [61]. Beyond hardware, autonomy requires robust software for signal processing, adaptive path planning, and system-level fault tolerance. Regulatory considerations such as FAA-imposed hardware constraints also play a pivotal role in aligning autonomous system design with safety and airspace integration standards [1]. Table 2 presents UAV autonomy levels with their key characteristics and commonly used systems and components.

Table 2
Levels of autonomy in UAVs.

Autonomy level	Key characteristics	Common systems/components
Manual	Human-in-the-loop, visual line-of-sight, no sensing	RC controller, direct pilot control
Teleoperated	Sensor feedback enabled, beyond-line-of-sight, no autonomy	Cameras, radio telemetry, IMU, human supervision
Semi-autonomous	Waypoint-based or supervised control, some obstacle avoidance	GNSS, IMU, LiDAR or stereo vision, path planning module, medium processor, system-level safety checks
Fully Autonomous	Real-time decision-making, obstacle avoidance, adaptive behaviors	LiDAR, depth camera, GNSS+INS/AHRS, onboard processors (e.g., NVIDIA Jetson), AI-based autonomous navigation

2.3.2. Map-based and mapless navigation

Autonomous navigation in UAVs can be broadly divided into *map-based* and *mapless* approaches, each leveraging different strategies to enable safe operation in environments. *Map-based navigation* depends on building and maintaining explicit environmental models through methods such as Simultaneous Localization and Mapping (SLAM) either offline or in real time. Using sensor fusion from SLAM, LiDAR, cameras, and inertial measurement units (IMUs) allows UAVs to construct probabilistic maps while simultaneously estimating their location. Algorithms like ORB-SLAM (for vision-based systems) and Cartographer (for LiDAR-based systems) are commonly used [7]. These methods perform well in structured environments by enabling accurate localization and global path planning using algorithms like A* or Rapidly-Exploring Random Trees (RRT) [8,9]. However, as the environment grows in scale or becomes more complex, maintaining the map for handling loop closures and global consistency can become computationally expensive. In contrast, *mapless methods* eliminate the need for intermediate map representations by directly mapping raw sensory input to control actions. These systems often adopt reactive or learning-based paradigms, such as end-to-end deep learning models that transform sensor data into control commands [62]. By focusing on real-time perception and action, these approaches often use lightweight techniques to optimize behavior under uncertainty. Their adaptability and reduced computational demands make them especially attractive for fast-changing or unstructured environments where traditional mapping may be infeasible.

Mapless methods are preferred in highly complex environments due to the limitations of map-based approaches in these settings. Dynamic obstacles, such as moving vehicles, pedestrians, or wildlife in urban or forested areas, require frequent map updates to ensure accuracy. Static assumptions in SLAM can cause errors like ghosting or drift when features change quickly [17]. For example, traditional SLAM techniques (e.g., visual or LiDAR-based) perform poorly in non-stationary scenes as they demand real-time detection, segmentation, and remapping of mobile elements. The process places heavy computational loads on UAVs with limited resources which causes latency and shorter flight times. It can also fail in GPS-denied or cluttered areas where partial observability affects mapping accuracy. Mapless methods address these challenges by relying on immediate sensory inputs rather than persistent maps, which allows agile responses through data-driven learning that adapts to new dynamics without comprehensive mapping. In DRL applications, mapless navigation integrates effectively with agents learning policies from high-dimensional inputs like camera feeds. These policies are first optimized for reward in simulated dynamic environments before being transferred to real-world applications, and they address challenges such as wind disturbances and occlusions more effectively than traditional map-based systems [63–65]. Although map-based methods provide reliability in predictable environments, their inflexibility amid uncertainty, such as rapid environmental changes or sensor noise, renders mapless alternatives more suitable for real-world autonomy. Overall, this trend shows a shift toward hybrid or fully mapless methods in UAV research, which aim to balance computational efficiency with reliable performance in uncertain, complex environments.

2.3.3. Path planning and trajectory planning

Path planning and trajectory planning serve as key elements in autonomous UAV navigation. They allow vehicles to create workable routes and movements that respect environmental limits, vehicle capabilities, and mission goals. Path planning aims to build a collision-free route from a start point to a target. It often sets aside time and dynamics in the early stages. It relies on algorithms like A*, Dijkstra, or sampling-based options such as Rapidly-exploring Random Trees (RRT) and its forms (for example, RRT*). These methods explore state spaces presented as grids, graphs, or probabilistic roadmaps [38,66]. In addition, trajectory planning improves these routes into time-linked paths. These paths consider kinematics, dynamics, and restrictions like speed, acceleration, and energy use. It applies tools such as polynomial splines [67] and Model Predictive Control (MPC) [68] to achieve smoothness and practicality. These steps form a core part of the navigation system. They connect with sensing and control units to support activities from waypoint tracking to advanced operations.

Path planning and trajectory planning are key to maintaining adaptability and safety in highly complex environments such as those with moving obstacles, wind disturbances, limited visibility, or areas without GPS. Dynamic path planning often relies on real-time replanning methods. Examples include A* and the dynamic window approach [69,70], which use local sensor data for reactive avoidance, and informed RRT*, which can quickly generate a safe path and then refine it over time [71]. These methods can require significant computation, particularly when frequent replanning is needed, and they may create less-than-optimal or oscillating paths in areas with many obstacles. Their effectiveness also depends heavily on the precision and refresh rate of onboard sensors. Trajectory planning adds to this by focusing on multiple goals. These goals cover low energy use [72], reduced collision risks through chance-constrained MPC [73], and strong responses to disturbances via feedback loops or random models. For example, in urban canyons or forested lands, these methods allow UAVs to handle narrow paths or avoid sudden dangers. They do this by regularly refreshing trajectories from blended sensor inputs, such as LiDAR point clouds or camera streams. This lowers failure rates in changing conditions [74]. More advanced methods integrate machine learning, including neural networks for predicting the movement of obstacles and DRL for developing end-to-end navigation policies. Such methods enable proactive decisions that traditional methods may overlook.

It is important to note that path and trajectory planning are used in both map-based and mapless navigation methods. As discussed earlier, map-based methods use explicit maps (e.g., those generated through SLAM) to plan globally optimal routes in well-structured environments, while mapless methods mostly rely on raw sensor data to perform local planning in unmapped or rapidly changing areas. Some systems combine these two ideas, for example, by using graph-based path finding together with sensor-driven trajectory smoothing [70,73,74]. This hybrid approach increases flexibility and allows UAVs to adapt to many tasks, including search-and-rescue or delivery in unstable conditions. As UAV autonomy develops, research continues to focus on algorithms that are both scalable and computationally efficient, so they can handle more complex environments and work smoothly with DRL for highly adaptive navigation.

2.3.4. Reactive UAV navigation

Reactive navigation in UAVs serves as a method for real-time, sensor-based decisions. It allows UAVs to react right away to environmental changes without depending on ready-made maps, deliberation or wide global planning [75,76]. This method focuses on local sensing and quick actions from sensor data. It produces fast control responses with lower computing demands [77]. Reactive navigation ties closely to mapless methods. It avoids the need for complete environment models and relies on raw sensor data to create control outputs directly or with moderate processing. In mapless systems, reactive strategies help UAVs move through unknown or unstructured areas by relying on instant sensory feedback, such as optical flow or proximity measurements, instead of building and maintaining maps. This connection makes reactive navigation a key part of autonomy that supports fast decision-making and improves agility, while also reduces computational load in situations where traditional mapping is not practical.

Reactive navigation is well-suited for operation in highly complex environments. Deliberative approaches often struggle in fast-changing settings like city streets with moving vehicles, forests with shifting branches, or disaster sites with unpredictable debris, This is because they require constant remapping, complete modeling regarding the environment, which takes significant computing power and time, and this can lead to delays, navigation errors, or even collisions. Reactive methods take a different approach by focusing on quick responses. They let UAVs adjust immediately to new conditions, such as a sudden obstacle or a strong wind gust, which helps maintain both safety and efficiency [76]. For example, in GPS-denied indoor spaces or in crowded outdoor areas, reactive algorithms can combine real-time sensor data to keep the UAV stable and moving toward its goal [78]. Such features make them suitable for real-time tasks like search-and-rescue or delivery in unstable settings. This stress on reaction instead of detailed planning fits the requirements of complex environments.

In reactive navigation, methods are often grouped into two main types: *end-to-end* and *non-end-to-end*. Each processes inputs and generates outputs differently, with variations in sensor-based, vision-based, or hybrid designs [4,26,79].

End-to-end approaches combine perception, decision-making, and control in a single model. They remove the need for a separate control unit by mapping raw inputs directly to UAV commands. In these systems, feature representations are not tied closely to specific environmental details but are directly linked to control actions such as velocity changes or attitude adjustments. For example, a CNN can encode features from sensor data and output roll, pitch, or yaw commands without intermediate processing.

Non-end-to-end approaches separate perception from control. They first create an environmental representation from the field of view, then pass this to a control unit. These representations are more closely tied to environmental structure extracted using CNNs or optical flow algorithms such as the relative positions of corridors or objects. The control unit then converts this data into commands, sometimes using distance-to-collision information formatted as matrices, which can be compressed through clustering or classification for efficient use.

In both approaches, most sensor-based variants rely on LiDAR, ultrasonic, or infrared sensors to estimate proximity and velocity, which supports reliable navigation in low-visibility conditions [80]. In contrast, Vision-based implementations often use RGB or depth camera inputs, processed by CNNs for obstacle detection and path correction, including monocular setups for small UAVs [26]. In addition, hybrid systems combine visual and sensor data, such as IMU readings, to improve reliability in varied environments [26,80]. End-to-end methods can learn complex, adaptive behaviors from high-dimensional inputs, which make them powerful in non-stationary environments. However, they may face challenges in interpretability and risk overfitting without large datasets. Non-end-to-end methods provide clearer debugging and greater adaptability, but may introduce slight delays. Therefore, they are often more suitable when explicit environmental representation is a priority. Both are capable of achieving strong performance in complex environments, and primarily leverage DRL to reduce latency and improve generalization to new scenarios.

3. DRL for autonomous UAV navigation in complex environment

This section discusses the application of DRL for autonomous UAV navigation, with a focus on addressing the challenges posed by complex environments through innovative algorithmic adaptations and frameworks. By examining core components, environmental factors, and specialized DRL designs, it aims to provide a comprehensive understanding of how DRL enhances UAV efficiency and robustness in real-world scenarios. Section 3 reviews DRL approaches for autonomous UAV navigation in complex environments. It first characterizes the key environmental factors that shape navigation difficulty, and then examines representative DRL methods for three major scenarios: cluttered spaces, dynamic obstacles, and wind disturbances. The discussion emphasizes how algorithm design, state representation, temporal information, exploration, experience reuse, and safety mechanisms are adapted to different environmental challenges. Finally, the section summarizes the major DRL design choices and highlights how these choices vary across different types of complex environments. Fig. 1 summarizes DRL design patterns among the studies reviewed in this section.

3.1. UAV navigation environment

In autonomous UAV navigation, understanding the characteristics and complexities of the operational environment is fundamental to designing reliable and adaptive algorithms. The effectiveness of navigation strategies for DRL depends heavily on how well the learning agent perceives, models, and reacts to its surroundings. Since UAVs often operate in uncertain and changing conditions, a precise understanding of environmental dynamics enables the development of policies that are both robust and generalizable.

One prior survey classifies UAV navigation environments using four modeling settings: *static-known, dynamic-known, static-unknown, and dynamic-unknown* [39]. In a *static-known* environment, the map is available and obstacles remain fixed, so the planner can rely on complete, stable scene information. A *dynamic-known* environment still provides overall knowledge of the space, but some obstacles or agents can move, so the planner must account for time-varying collisions and re-planning. A *static-unknown* environment raises the difficulty because obstacles are fixed yet their locations (and sometimes even the environment scale) are not given in advance, which pushes the system toward online mapping, exploration, or reactive ‘‘sense-and-avoid’’ behavior. The most demanding case is the *dynamic-unknown* environment, where the agent faces both incomplete knowledge and moving obstacles, leading to partial observability, non-stationary dynamics, and frequent distribution shift. Since this survey focuses on mapless autonomous UAV navigation in complex environments, our emphasis naturally falls on partially or fully unknown and highly dynamic settings, which best reflect real deployment constraints.

Another common higher-level view classifies environments by spatial domain, including *indoor, outdoor (urban/suburban), and natural scenes* [81]. These domains present unique operational challenges and implications for learning-based control. *Indoor environments* are typically structured and confined spaces, such as warehouses, industrial facilities, or office buildings. These settings often involve narrow corridors, obstacles with well-defined boundaries. In such environments, DRL-based navigation focuses on local obstacle avoidance, precise maneuvering, and reward shaping for efficient exploration in maze-like layouts [82]. *Outdoor environments* are semi-structured and cover large-scale areas such as urban or suburban regions. They may range from dense urban canyons filled with moving vehicles and pedestrians to more open suburban fields. Key challenges include dynamic obstacles, variable weather conditions, and the need for long-range path planning that adapts to external changes in real time [83]. These conditions demand robust DRL frameworks that integrate perception, prediction, and control for continuous adaptation. *Natural environments* are largely unstructured and include forests, mountains, coastal

regions, and other uneven terrains. These domains present complex topographies, unpredictable wind fields, vegetation interference, and visibility fluctuations from fog, rain, or lighting changes. As a result, DRL algorithms must emphasize terrain-aware decision-making, adaptive control, and exploration under uncertainty [75,84].

While the spatial classifications offer a useful view, they are often too broad to explain why a DRL navigation policy succeeds in one setting and fails in another. We propose a more practical way to analyze DRL-based navigation by describing the environment along three key dimensions: *obstacle density, obstacle dynamics, and environmental disturbances*. *Obstacle density* spans from low-density open spaces, where planning is mostly about efficiency, to cluttered obstacle-rich scenes, where safe motion requires tight maneuvering and strong perception. Since this survey targets complex environments, we will emphasize cluttered settings in this study. *Obstacle dynamics* span from static to dynamic layouts where obstacle's motion varies over time quickly. Dynamic obstacles create non-stationary transition patterns and introduce the need for prediction and risk-aware decision-making, so we focus on these cases as well. In practice, the same policy that works in static maps can break when obstacle motion changes speed or direction, which makes dynamic benchmarks essential for fair evaluation. *Environmental disturbances* include wind, weather effects, and sensor noise, all of which can degrade both state estimation and control stability. Because disturbances are hard to summarize with a single scalar, it is often clearer to model them as separate factors. In this survey, we primarily focus on wind disturbances, as they represent the most common environmental perturbation in UAV deployments and have a direct impact on navigation performance and operational safety. Taken together, these three dimensions provide a clear and structured framework for organizing prior work in this context. More specifically, in the following subsections we narrow our discussion to DRL algorithms and their adaptations for three representative types of complex navigation scenarios: (i) navigation in cluttered spaces, (ii) navigation with dynamic obstacles, (iii) navigation under wind disturbance.

3.2. Navigation in cluttered spaces

Cluttered environments present a fundamental challenge for mapless UAV navigation because the vehicle must simultaneously make progress toward a destination, maintain sufficient clearance from nearby obstacles, and avoid becoming trapped in locally difficult geometries such as narrow passages, dead ends, and densely packed obstacle fields. Unlike navigation in relatively open spaces, successful flight in clutter depends not only on the underlying reinforcement learning algorithm, but also on how environmental information is represented, how useful experiences are collected, and how exploration is guided during training. Existing studies therefore show a gradual shift from simply applying standard deep reinforcement learning algorithms toward designing perception, memory, experience-reuse, and exploration mechanisms specifically for cluttered navigation. A summary of the works discussed in this section is provided in Table 3.

3.2.1. Overview of DRL approaches for cluttered navigation

A range of deep reinforcement learning algorithms have been adopted for UAV navigation in cluttered environments, with the choice largely determined by the form of the action space and the required degree of control flexibility. Early studies frequently employed Deep Deterministic Policy Gradient (DDPG), which is well suited to continuous actions such as velocity, heading, altitude, and thrust. For example, Omar et al. demonstrated continuous three-dimensional obstacle avoidance and introduced transfer learning from an obstacle-free task to accelerate learning in cluttered scenes [87]. Subsequent work improved DDPG by addressing unstable value estimation and competing navigation objectives. The state-decomposition DDPG method separately processes obstacle-related and goal-related information before producing the final control action [85], while multicritic-delayed DDPG employs multiple critics and delayed policy updates to improve learning stability in dense obstacle fields [86].

TD3 has become another widely used choice because its twin critics and delayed policy updates reduce some of the overestimation and instability associated with standard DDPG. Its use in cluttered navigation is commonly accompanied by additional learning mechanisms. For instance, Demonstration-assisted TD3 initializes both the policy and critic using trajectories generated by a conventional planner, which reduces the amount of unsafe exploration needed at the beginning of training [91]. Other TD3-based approaches incorporate recurrent memory, attention, or privileged information to improve decision-making when observations are incomplete or noisy [30,88].

Soft Actor-Critic (SAC) provides a stochastic alternative that explicitly encourages exploration through entropy maximization. This characteristic is useful in cluttered spaces, where the UAV may need to discover detours rather than repeatedly follow the most direct route. SAC has consequently been combined with curriculum learning for narrow-gap traversal [100], Hindsight Experience Replay for goal-conditioned navigation [105], and intermediate-target mechanisms for escaping dead ends and maze-like structures [106]. PPO has also been adopted for mapless navigation, particularly when stable policy updates are desired. Representative studies use PPO for continuous local motion planning from LiDAR observations [102] and for navigation based on a small number of range sensors [99].

In contrast to these continuous control actor–critic methods, Deep Q-Networks (DQN) are primarily used when navigation is represented through a limited set of discrete actions, such as moving forward or turning left and right. Their relatively simple action formulation makes them attractive for high-level navigation and perception-oriented tasks. For instance, memory-enhanced DQN combines visual observations, recent actions, and goal information to reduce repeated turning and cyclic trajectories around large obstacles [92]. Other DQN-based methods integrate object detection features or local map information to improve obstacle awareness while keeping the state representation compact. Recurrent variants such as DRQN further incorporate temporal information, which enable the policy to reason over recent observations when a single frame is insufficient [17].

The literature reviewed in this study suggests that no single DRL algorithm consistently dominates cluttered UAV navigation. More importantly, performance in highly cluttered environments increasingly depends on how these algorithms are complemented by appropriate state representations, memory, experience reuse, and exploration strategies.

3.2.2. State representation and perception under clutter

A central design question is how much information the UAV requires to make reliable local decisions. Earlier approaches often rely on compact state vectors containing UAV motion variables, goal-relative information, and a limited number of range measurements. Such representations have the advantage of low computational cost and can be effective when obstacle geometry is relatively simple. The state-decomposition DDPG method, for example, uses twelve range measurements together with flight and goal information, but processes obstacle-related and navigation-related states separately [85]. This decomposition reflects an important insight: in dense clutter, obstacle avoidance and goal seeking can compete with one another, and treating all input variables identically may obscure their different roles.

Several studies therefore explicitly separate perceptual information from navigation information. The adaptive control TD3 framework introduces attention that changes emphasis between obstacle-related sensor inputs and goal-related features according to the immediate situation [30]. Similarly, hierarchical and attention-based navigation methods combine depth images with goal distance and heading information while using recurrent networks and temporal attention to

Table 3
Representative studies of navigation in cluttered spaces.

Ref	Year	Targeted application & environment	Base RL	Control dimension	Simulation & test
[85]	2023	3D point-to-point navigation in dense static and refreshed dynamic clutter	DDPG	3D continuous	1800 × 1800 m randomized 3D obstacles; Monte Carlo tests
[86]	2022	Online goal-reaching in unknown cluttered environments with static obstacles	DDPG	2D continuous	Synthetic 2D square world with cylindrical obstacles and range sensors
[87]	2020	3D urban navigation to static or moving targets with obstacle avoidance	DDPG	3D continuous	Python 3D simulator; 100 m cubic urban scenes with polygon obstacles
[30]	2024	Low-altitude point-to-point navigation in complex valley terrain with static obstacles	TD3	3D continuous	UE4 3D valley world with lakes, jungle, mountains, cylindrical obstacles
[88]	2024	Vision-based high-speed navigation in unknown clutter under partial observability	TD3	3D continuous	UE4 + AirSim; cylindrical maze, cubic maze, forest, mixed terrain
[89]	2024	Sparse-reward exit navigation in cluttered indoor 3D rooms	TD3	3D continuous	Microsoft AirSim indoor rooms with increasing obstacle density
[90]	2022	Aggressive flight through narrow windows and slalom obstacles	TD3	3D continuous	UE4 + AirSim simulation; real Hummingbird quadrotor with Vicon
[91]	2020	Local obstacle avoidance in unknown cluttered 3D environments	TD3	3D continuous	ROS/Gazebo PX4 SITL; AirSim Rocks and Neighborhood
[92]	2022	Vision-based fixed-altitude goal navigation in occluded realistic 3D scenes	DQN	2D discrete	Microsoft AirSim Blocks and Neighborhood scenes
[17]	2021	Local navigation among many moving obstacles in dynamic low-altitude airspace	DRQN	2D discrete	2D 500 × 500 planar simulator with dynamic obstacles
[93]	2020	Indoor goal-directed navigation with monocular sensing and static obstacles	DQN	2D discrete	ROS-based 20 × 20 planar simulation across multiple indoor scenes
[94]	2020	Grid-based disaster target search and sequential hotspot finding	DQN	2D discrete	2D Snake simulation; limited 12 × 12 outdoor grid tests
[95]	2019	Outdoor exploration and target-reaching in cluttered partially observable environments	DQN	2D discrete	AirSim forest, weather, plain-field, and savanna scenarios
[96]	2023	Agile narrow-gap traversal with varying gap roll angles	SAC	3D continuous	Custom simulator, Gazebo PX4 SITL, indoor real-world
[97]	2023	Vision-based goal navigation in unknown cluttered static 3D environments	SAC/DrQ	2D continuous	AirSim with 3 environments; preliminary DJI F550 indoor tests
[98]	2023	High-speed FPV gate racing under perception and dynamics uncertainty	PPO	3D continuous	Real 7-gate races; high-fidelity quadrotor simulator
[99]	2023	BVLOS-style goal navigation in unknown sparse and dense static environments	PPO	3D discrete	AirSim 3D easy/hard areas; 2D grid pre-screening
[100]	2021	Aggressive tilted narrow-gap traversal with tight clearance	SAC	3D continuous	Custom simulator; real F330 quadrotor with Vicon
[101]	2021	Sensor-guided anomaly-source navigation in partially observable static grids	PPO	2D discrete	Unity ML-Agents 2D grid-world across obstacle densities
[102]	2021	GPS-denied outdoor point-goal navigation in forest-like static clutter	PPO	2D continuous	Gazebo simulation; real outdoor forest flight tests
[103]	2020	Indoor object-goal visual navigation in cluttered unseen environments	PPO	2D discrete	Habitat/Gibson simulation; Turtlebot3 and DJI Tello tests
[104]	2025	Vision-based navigation and agile traversal of rings and narrow gaps in cluttered 3D environments	TD3 + PPO	3D continuous	MATLAB/Unreal Engine 5 simulation; real quadrotor tests through rings and tilted narrow gaps
[105]	2023	Goal-oriented path planning and collision avoidance in static cluttered environments	SAC	3D continuous	3D simulation with randomized start–goal positions and nine static obstacles; simulated hexarotor trajectory tracking
[106]	2022	Mapless goal-reaching and local-minimum escape in maze-like cluttered environments	SAC	2D continuous	AirSim/Unreal Engine simulation across five training environments and one unseen maze; real TurtleBot3 sim-to-real test

identify which recent observations are most relevant for the current maneuver [97].

Vision-based methods provide richer geometric information but introduce greater perceptual complexity. Depth images are widely used because they directly encode obstacle structure without requiring a complete environmental map. The demonstration-assisted TD3 approach maps depth images and relative goal information directly to continuous velocity commands [91], while privileged-learning methods employ depth perception together with UAV self-state information and deliberately corrupt these observations during training to prepare the policy for noisy sensing [88]. Other studies use RGB imagery and learned visual features. Memory-enhanced DQN, for example, combines visual observations with recent actions and goal information to reduce cyclic behaviors when large obstacles block the direct route to the destination [92].

Memory becomes particularly important because local sensing rarely reveals the full geometry of a cluttered environment. A vehicle observing only its current depth image or range measurements may not know whether turning around an obstacle will eventually lead toward the goal or into a dead end. Recurrent networks, stacked observations, and explicit memory mechanisms therefore provide short-term context. Layered recurrent Q-learning uses LSTM-based networks

to distinguish obstacle avoidance from target acquisition [17], whereas memory-enhanced DQN explicitly stores recent visual and action information [92]. These approaches illustrate a broader trend that as clutter becomes more geometrically complex, successful navigation increasingly depends on remembering how the UAV arrived at its current state rather than reacting solely to instantaneous obstacle distance.

3.2.3. Learning strategies for efficient exploration and collision avoidance

Efficient exploration is particularly difficult in cluttered environments because random actions frequently produce collisions, while overly conservative policies may learn to stop, oscillate, or repeatedly choose familiar safe routes. Consequently, many recent contributions focus less on changing the basic RL algorithm and more on improving the quality of experience from which it learns.

Experience reuse is one important direction. Demonstration-assisted TD3 initializes both the policy and value estimator from trajectories produced by a conventional planner before allowing the UAV to improve through its own interaction [91]. This provides meaningful collision-free behavior early in training and reduces dependence on purely random exploration. Hindsight Experience Replay addresses a related problem from another perspective that unsuccessful trajectories may still contain useful navigation experience. The SAC-HER framework relabels states that the UAV actually reached as alternative goals, which allows failed episodes to generate additional successful training examples [105]. This is valuable in clutter because reaching the intended destination may initially be rare.

More recent studies extend hindsight beyond conventional goal relabeling. Hindsight intermediate targets introduce temporary goals when the vehicle appears trapped by a wall, corridor, or cul-de-sac. Importantly, the resulting detour experiences are subsequently learned in relation to the original destination, and it teaches the policy that temporarily moving away from the goal can ultimately be beneficial [106]. This is particularly important in cluttered navigation, where a simple "always move closer to the goal" strategy often fails. Sun et al. developed a different idea that instead of merely relabeling a recorded trajectory, previously reached states are used as alternative goals and relevant actions are re-executed to create physically consistent successful experiences [104]. Although this requires additional environmental interaction, it represents a notable attempt to make hindsight learning more suitable for vision-based UAV navigation.

Intrinsic motivation and entropy-based exploration provide another strategy. Lv et al. proposed an entropy explorer approach that rewards both visiting unfamiliar states and maintaining diversity in actions, substantially improving TD3 performance as obstacle density increases [89]. A different curiosity-driven approach for aggressive flight similarly rewards trajectories that differ from previously experienced ones, encouraging discovery of alternative maneuvering patterns rather than repeatedly following a narrow set of behaviors [90]. Curriculum learning complements these approaches by beginning with easier navigation conditions and gradually increasing difficulty. This principle has been used for narrow-gap traversal [100] and for navigation across increasing obstacle densities. This allows fundamental goal-seeking behavior to develop before the policy encounters highly constrained scenarios.

3.3. Navigation with dynamic obstacles

Dynamic obstacles introduce a fundamentally different challenge from static clutter because safe navigation depends not only on identifying where obstacles are located, but also on understanding how their positions may change during the UAV's response. A moving obstacle that is currently distant may quickly become dangerous, whereas a nearby obstacle moving away may require little intervention. Consequently, DRL-based navigation in dynamic environments has gradually evolved from purely reactive obstacle avoidance toward methods that incorporate temporal information, more effective learning from critical experiences, and safety-aware decision making. Table 4 provides an overview of the representative works discussed in this section.

3.3.1. Overview of DRL approaches in dynamic environments

Early studies commonly adopted value-based methods such as DQN and DDQN because of their relatively simple training procedure and compatibility with discrete navigation actions. Wang et al. [118], for example, decomposed navigation into obstacle-avoidance and goal-directed modules and used deep Q-learning to coordinate the two tasks. Their modular design demonstrated that DRL can learn collision avoidance in environments containing rapidly moving obstacles without requiring an explicit global map. Memory-enhanced DQN approaches further improved this idea by allowing past observations to influence current decisions [29]. However, discrete action spaces generally restrict the UAV to a small number of predefined maneuvers and may therefore be less suitable for smooth and agile flight.

For this reason, actor–critic algorithms supporting continuous control have become increasingly prominent. DDPG was one of the earliest widely adopted approaches. Wan et al. proposed Robust-DDPG, introducing delayed learning, adversarial state perturbations, and mixed exploration to improve stability and robustness under moving obstacles and environmental uncertainty [108]. Hu et al. subsequently developed REL-DDPG, in which experience selection is tailored to the UAV's current situation, improving convergence in complex three-dimensional environments [107]. Nevertheless, DDPG can suffer from unstable value estimation, which motivates increased adoption of TD3. Zhang et al. employed TD3 with a two-stream actor–critic architecture and showed that incorporating changes in environmental observations improves adaptation to moving obstacles [27]. Later TD3-based studies further incorporated prioritized replay and modified value estimation to improve training efficiency and stability [109].

SAC has also attracted attention because its stochastic exploration is useful in complex continuous environments. A comparative study by Kalidas et al. found SAC more effective than DQN and PPO in their vision-based dynamic navigation scenarios, particularly because it combines continuous control with replay-based learning and exploration [121]. More recent SAC methods have augmented the state with temporal information and adjusted reward priorities during training [119,120]. Meanwhile, PPO despite being less sample-efficient because it learns primarily from newly collected interactions, remains attractive because of its stable policy updates and has been successfully applied to highly dynamic real-world navigation. Recent systems using PPO have demonstrated avoidance of fast-moving obstacles and physical deployment [122,123].

Similar to the algorithms discussed in the last section, no single DRL family is universally preferable in dynamic environments. Increasingly, performance gains arise not simply from selecting a stronger base algorithm, but from how environmental dynamics are represented and how learning is organized around efficiency and safety-critical experiences.

3.3.2. Dynamic state representation and temporal perception

A central difficulty in dynamic environments is that a single observation describes obstacle position but generally provides insufficient information about motion. Consequently, state-space design has progressively shifted from purely spatial representations toward spatio-temporal representations that allow the policy to distinguish approaching, departing, and crossing obstacles.

One straightforward approach is to provide changes between consecutive observations. Wang et al. developed a two-stream Q-network that separately processes the current laser observation and its temporal difference, allowing the policy to infer obstacle movement without explicitly predicting trajectories [118]. A similar principle was later applied to continuous control by Zhang et al. [27], where a TD3-based architecture processes both the current environmental observation and its change. This simple design is important because it adds motion awareness while avoiding the computational cost of explicit tracking and prediction.

More complex environments, however, may require information over longer periods. Singla et al. [29] addressed this problem using recurrent memory and temporal attention. Rather than making decisions

Table 4
Representative studies of navigation with dynamic obstacles.

Ref	Year	Targeted application & environment	Base RL	Control dimension	Simulation & test
[107]	2021	Goal-reaching with obstacle/threat avoidance in unknown 3D static/dynamic environments	DDPG	3D continuous	Custom 3D fixed-wing simulator; 32-ray sensing; static layouts and one dynamic scenario
[108]	2020	Goal-reaching with static/moving threat avoidance in uncertain dynamic planar environments	DDPG	2D continuous	2D simulator, 400 × 300 m, stochastic moving threats, LiDAR sector sensing
[27]	2022	Point-to-point navigation in random static/dynamic 3D multi-obstacle environments	TD3	3D continuous	Unity3D ML-Agents; static training, dynamic testing, 2000-episode evaluation
[109]	2024	Point-to-point path planning in cluttered static and moving-obstacle 3D environments	TD3	3D continuous	Custom 3D OpenAI Gym; static/dynamic scenarios; compared with DDPG, TD3, SAC
[110]	2024	Vision-based point-to-point navigation in realistic static and dynamic 3D scenes	TD3	3D continuous low-level	AirSim with Unreal Engine; neighborhood and city environments; RGB/RGB-depth comparisons
[111]	2024	Local goal-reaching and obstacle avoidance in unfamiliar indoor-like environments	TD3	3D continuous high-level	Gazebo with PX4, HWT_OS, MAVROS; no-obstacle, static, dense, dynamic tests
[112]	2023	Fixed-wing low-altitude target-reaching in unknown dynamic 3D threat environments	TD3	3D continuous	Custom large-scale 3D fixed-wing simulator; moving obstacles, varied speeds/densities
[113]	2025	Urban low-altitude path planning with static/dynamic obstacles and wind	DQN	2D continuous	AirSim–physical UAV twin platform; urban tasks, prediction, wind tests
[114]	2024	High-speed, high-density dynamic 3D obstacle avoidance and goal-directed planning	PDQN	3D hybrid discrete-continuous	Custom OpenAI Gym 3D simulator; static/dynamic mixed-motion obstacle tests
[115]	2024	Vision-based goal-reaching and collision avoidance in unknown obstacle fields	DQN	3D discrete	Python SVFO training and Unity3D testing with static/moving obstacles
[116]	2020	Goal-directed navigation with static/dynamic obstacles and multi-UAV interaction	DDQN	2D discrete	2D grid simulation; randomized training, deterministic transfer tests
[117]	2019	Quadrotor goal-reaching in suburban clutter with static and moving drones	DDQN	2D discrete	AirSim/Unreal; fixed-altitude urban neighborhood; 100-episode tests with 0–2 moving drones
[29]	2018	Indoor local obstacle avoidance with static clutter and moving humans	DRQN	2D discrete	Gazebo indoor environments; simulated RGB-D datasets; six test scenes including dynamic cafe
[118]	2018	Dynamic indoor goal-reaching among moving obstacles using laser and goal vector	DQL	2D discrete	Custom 2D simulator; four rooms; randomized training, deterministic testing
[119]	2025	Local goal-reaching in dense static/dynamic obstacle fields, including concave traps	SAC	2D continuous	2D Gym environment; static/dynamic obstacle tests varying count, size, speed
[120]	2024	Fixed-altitude point-to-point navigation in high-density, highly dynamic clutter	SAC	2D continuous	Gazebo; dense static and moving-obstacle planar scenarios with LiDAR
[121]	2023	Vision-based exit-reaching in static and dynamic 3D cluttered environments	SAC/PPO/DQN	2D discrete/continuous	AirSim/Unreal; three training worlds, two test worlds, moving actors and obstacles
[122]	2025	Real-time quadrotor navigation in highly dynamic clutter with fast-moving obstacles	PPO	2D continuous	Unity ML-Agents plus real quadrotor tests with lidar and PX4
[123]	2025	Safe goal-reaching in cluttered dynamic spaces with dense static and moving obstacles	PPO	3D continuous	Isaac Sim training, Gazebo evaluation, real quadcopter flight with pedestrians
[124]	2026	Mapless goal-reaching and obstacle avoidance in unknown cluttered environments with static and moving obstacles	SAC	3D continuous	AirSim simulation across playground, canyon, and dynamic urban scenarios; indoor real-UAV test
[125]	2025	Goal-directed UAV navigation and dynamic obstacle avoidance in cluttered indoor environments with moving pedestrians	DQN	3D discrete	AirSim/Unreal Engine 4 simulation with varying obstacle densities
[126]	2025	3D local path planning and obstacle avoidance in complex post-disaster environments with static and moving obstacles	TD3	3D continuous	Large-scale 3D simulation with dense, dynamic, and multi-UAV scenarios; real-flight validation

solely from the current monocular observation, the network retains information from previous observations and emphasizes temporally relevant features. This is particularly beneficial when obstacles become temporarily occluded or when their direction of movement cannot be determined from two consecutive frames. This idea is developed further by integrating a GRU and temporal attention mechanism with SAC [119]. Historical states are encoded and selectively weighted so that the policy can preserve useful motion information while remaining responsive to current conditions. Such designs effectively acknowledge that mapless navigation with limited sensors is often partially observable, even when the problem is formally described as an MDP.

Recent approaches have also moved toward more explicit descriptions of obstacle dynamics. Sheng et al. [120] augment LiDAR states with changes in position and orientation, while NavRL separately represents dynamic obstacles using their relative positions, dimensions, and estimated velocities [123]. Fan et al. took a different approach by stacking historical LiDAR scans into a compact obstacle map, enabling a neural network to extract both spatial structure and motion patterns directly [122]. Vision-based frameworks similarly combine depth information with UAV and goal states to improve awareness of surrounding geometry [110]. Collectively, these studies reveal a clear trend that successful dynamic navigation increasingly depends on representing not only where an obstacle is, but also how it is moving and how its motion relates to the UAV's future trajectory. Temporal difference features offer a lightweight solution, whereas recurrent and attention-based models capture longer dependencies. Moreover, explicit velocity estimation and historical sensor data provide richer motion information for highly dynamic environments

3.3.3. Learning strategies for efficiency and safety-critical policy design

Dynamic obstacle avoidance also places substantial demands on the learning process and strategies. Collisions and near-collision situations may represent only a small proportion of all collected interactions, yet they contain some of the most valuable information for learning safe behavior. This has motivated considerable research on experience replay. REL-DDPG selects experiences that are not only associated with high learning error but are also relevant to the UAV's current navigation situation [107]. Hu et al. later proposed asynchronous curriculum experience replay, in which experience priorities are continuously updated and samples are introduced from easier to more difficult situations as training progresses [112]. Similarly, prioritized experience replay has been combined with TD3 to increase the frequency with which informative transitions are revisited [109,110]. Instead of treating all past interactions as equally useful, these methods direct learning toward experiences that carry greater information about difficult or safety-critical situations in dynamic environments.

Reward design provides another important mechanism for introducing safety knowledge into the learning process. Most approaches combine progress toward the destination with penalties for collision and unsafe obstacle proximity, but more recent studies increasingly incorporate structured prior knowledge rather than relying entirely on trial-and-error learning. For instance, Sheng et al. [120] dynamically adjust reward weights throughout training, allowing broader exploration initially and progressively emphasizing collision avoidance and goal-directed motion. A different approach incorporates obstacle motion into their reward formulation so that the UAV is discouraged from entering potential future collision regions rather than reacting only to instantaneous obstacle distance [122]. Artificial potential fields (APFs) provide a particularly intuitive means of introducing such guidance. Liu et al. incorporated APF-based attraction and repulsion into the reward function, explicitly encouraging motion toward the target while increasing penalties as the UAV approaches obstacles [125]. A similar approach integrates APF-derived repulsive information into the reward of a SAC-based visual navigation policy [124]. Instead of considering only the nearest obstacle, their reward aggregates the influence of multiple nearby obstacle points, providing a denser indication of surrounding collision risk.

Beyond reward shaping, recent studies increasingly incorporate explicit mechanisms that constrain or correct unsafe policy outputs. A tighter integration between model-based guidance and DRL is adopted by Ren et al. [126]. Their PG-ITD3 method does not ask the policy to directly determine UAV motion. Instead, TD3 adjusts the repulsive coefficients of an improved APF, which subsequently determines the navigation direction. The APF therefore provides an explicit collision-avoidance structure while DRL adapts its parameters to changing obstacle configurations. This illustrates a broader trend toward embedding task-specific safety knowledge into both the policy structure and the learning process. For instance, NavRL applies a lightweight safety shield based on velocity-obstacle principles [123]. This approach uses a PPO policy to proposes a velocity that may lead to a collision, the command is projected toward a safer alternative. Related hybrid approaches use APFs, predicted obstacle motion, or structured collision-risk models to guide learned decisions [124–126]. Such designs reduce the burden on the DRL policy to discover every aspect of safe navigation solely through accumulated rewards.

3.4. Navigation under wind disturbance

Navigation under wind disturbances and perception uncertainty remains one of the central challenges in autonomous UAV systems operating in complex environments. These factors directly influence system stability and mission reliability and are therefore a primary focus of this survey. Accordingly, this survey focuses on approaches designed to address these challenges. Wind can push the vehicle away from its intended path, reduce tracking accuracy, and cause unsafe behavior if the controller cannot respond quickly. At the same time, limited perception and sensing can prevent the UAV from correctly understanding its surroundings. Recent DRL studies address these issues at different levels of the autonomy stack. Some focus on low-level control and tracking under external disturbances, others study disturbance-aware replanning and mapless navigation in uncertain environments. At the control level, wind directly perturbs vehicle dynamics, making accurate hovering, trajectory tracking, and disturbance rejection more difficult. At the navigation level, wind can alter the feasibility, safety, and efficiency of planned motions, requiring the UAV to adapt its path or local navigation policy to changing conditions. To clarify these developments, this section reviews the literature from these perspectives. The studies covered in this section are summarized in Table 5.

3.4.1. Wind-robust low-level control and tracking

A major area of research addresses wind disturbance at the low-level control layer, where the main goal is to keep the UAV stable and accurate despite external forces. Rather than solving full path planning, these studies focus on core flight behaviors such as hovering, waypoint regulation, and trajectory tracking under changing wind conditions. A common pattern is to strengthen the control layer rather than redesign the full navigation stack. Several works combine reinforcement learning with classical control ideas. For example, Ma et al. [129] adds a DDPG-based compensation term on top of a dual-loop PID controller, allowing the UAV to better reject composite wind effects while improving hover and trajectory tracking. In a similar spirit, Pi et al. [64] introduces a disturbance-compensation framework in which a pre-trained RL controller is supported by an external observer that estimates wind-like forces and corrects the control signal, showing that robustness can be improved even without retraining the policy under disturbances. Likewise, an integral compensation mechanism is combined with DDPG learning in [134], directly targeting the steady tracking errors that often remain under persistent disturbance. Another line of work improves robustness through training design. REFORMA [128] frames disturbance rejection as a robustness problem and trains a policy with adversarial disturbances, while also adding an adaptation module that

Table 5
Representative studies of navigation under wind disturbance.

Ref	Year	Targeted application & environment	Base RL	Control dimension	Simulation & test
[127]	2025	Target search, continuous tracking, obstacle avoidance in cluttered occluded environments	DDPG	2D continuous	2D Gym-style grid world, polygon obstacles, occlusion zones, noisy sensing
[128]	2024	Robust hovering and gate traversal under unknown action disturbances	PPO	3D continuous	gym-pybullet-drones; 3D hover and fly-through-gate under adversarial disturbances
[129]	2024	Wind-disturbance rejection for hover and trajectory tracking	DDPG	3D continuous	MATLAB/Simulink; composite wind, hover, lawnmower tracking, limited real-flight tests
[130]	2024	Point-to-point regulation and hovering under wind and noise disturbances	A2C/SAC/ PPO	3D continuous	Custom Gym-Bullet quadrotor simulator with multiple synthetic wind/noise models
[131]	2023	Local replanning with collision avoidance under wind disturbances	PPO	3D continuous	ROS, Gazebo, Rotors; 3D power-plant environments with sparse, dense, maze zones
[19]	2023	Trajectory/target tracking under time-varying wind disturbances	DDPG	3D continuous	Python/TensorFlow simulation, AirSim HITL, outdoor quadrotor flight tests
[132]	2023	Stand-off target tracking for tilt-rotor VTOL under wind	DDPG	3D continuous	Numerical simulation, HIL platform, real-world VTOL tests
[64]	2021	Hovering and waypoint tracking under external wind disturbances	PPO	3D continuous	Python simulator, indoor fan-wind tests, outdoor GPS square-path flights
[133]	2021	Robust waypoint reaching and hovering under disturbances and model mismatch	DDPG	3D continuous	MuJoCo Gym simulation; unseen mass and disturbance generalization tests
[134]	2020	Robust low-level tracking under wind, faults, and uncertainty	DDPG	3D continuous	High-fidelity quadrotor simulator with robustness and online-adaptation tests
[135]	2024	Urban navigation with obstacle avoidance and wind-zone-aware goal-reaching	PPO + LSTM	3D continuous	Unity3D ML-Agents; random districts; Manhattan CFD-wind validation
[113]	2025	Urban goal-directed path planning with dynamic obstacles and wind	DQN + LSTM	3D discrete	AirSim-based digital twin; virtual-real experiments; static, dynamic, windy scenarios
[136]	2021	Point-to-point and waypoint navigation with static obstacles and wind	TD3/SAC	3D continuous	ROS, Gazebo, RotorS; 3D indoor scenarios with wind disturbances
[108]	2020	Point-to-point flight with collision avoidance in dynamic uncertain environments	DDPG	2D continuous	2D simulator; moving obstacles, density sweeps, disturbance robustness tests

helps the controller respond to different disturbance strengths during deployment. This is important because real wind conditions are not fixed and often change over time. Along related lines, the work in [133] applied action-robust DDPG to quadcopter control, using adversarial action perturbations during learning so that the final controller remains more reliable under model mismatch and control uncertainty. The approach in [130] takes another route by reshaping the learning signal through multiple PID-based evaluators and a self-attention mechanism, with the goal of making training more stable and the resulting controller more resistant to noisy or changing disturbances. A third group emphasizes tracking performance under changing wind conditions. The method in [19] improves trajectory tracking by combining incremental adaptation, guided exploration, and weighted learning updates, which helps the DDPG-based controller adjust when wind conditions shift across operating regimes. The study in [132] extends this idea to target tracking for a tilt-rotor UAV, highlighting a quantum-inspired replay strategy and disturbance-aware training to improve tracking stability in windy conditions.

3.4.2. *Wind-aware replanning and mapless navigation*

Wind-aware local replanning and mapless navigation studies move beyond basic stabilization and focus on how a UAV should make navigation decisions when wind changes the safety and feasibility of different routes. In this aspect, the main challenge is not only to resist disturbance, but also to adjust motion decisions online so that the vehicle can still reach its goal efficiently and safely. An earlier study [108] on robust point-to-point navigation in uncertain environments emphasizes a simple yet important idea: robustness can be improved by training policies under disturbances and challenging conditions, such as moving obstacles and noisy motion. Through exposure to these scenarios during training, the learned policy develops behaviors that remain effective even as environmental conditions deteriorate. Another representative example is the meta-learning replanner in [131], which adjusts local motion plans around a given reference trajectory and explicitly changes timing and path corrections when wind and obstacle density vary; its key contribution is fast adaptation to new disturbance regimes with only a small amount of online adjustment. For mapless urban navigation, Wu et al. [135] demonstrates that wind zones around buildings can generate traps and detours even when the destination is nearby. The study introduces a multi-objective formulation that balances goal reaching, obstacle avoidance, and wind exposure by leveraging compact perception features and temporal memory. In dynamic and cluttered environments, the approach in [113] emphasizes system-level integration. The framework combines obstacle motion prediction, a safety-shaping layer, and reinforcement learning, and is supported by a digital-twin loop that uses virtual–physical feedback to enrich training data and narrow the gap between simulation and real-world deployment. Another study [136] observes that the combination of sparse sensing and wind disturbances makes navigation strongly dependent on historical information. By integrating memory and stabilizing value estimation, the method achieves improved performance in 3D goal-reaching and waypoint navigation tasks under wind conditions.

Meanwhile, Feng et al. [127] presents a related perspective through its investigation of target search and continuous tracking in cluttered environments. Although the focus is on occlusion and temporary target loss rather than wind, the proposed search–track–relocalize strategy illustrates an important insight that effective local navigation policies should incorporate explicit recovery behaviors to handle disruptions caused by environmental uncertainty such as wind.

3.5. DRL design choice in complex environments

The design of a deep reinforcement learning system for UAV navigation should be closely matched to the type of complexity present in the operating environment. Although similar DRL algorithms may be used across different scenarios, the main challenge is not simply choosing a particular algorithm, but determining what information the UAV must understand and what behavior it must learn.

In cluttered spaces, the dominant difficulty is spatial, where the UAV must perceive nearby obstacles, identify feasible free space, and continue progressing toward the goal without becoming trapped in narrow passages, dead ends, or dense obstacle arrangements. As a result, design choices in these environments place strong emphasis on spatial perception and efficient exploration. State representations commonly focus on obstacle distance, depth information, local geometry, and relative goal direction, while memory can help the UAV avoid repeatedly visiting the same unfavorable regions. Because successful trajectories may be difficult to discover when collisions occur frequently, the learning process also benefits from mechanisms that make better use of limited experience. Demonstration-based learning, hindsight learning, curriculum learning, and exploration incentives can all reduce ineffective trial-and-error and help the policy discover useful detours. The main design objective in cluttered environments is therefore to help the UAV understand spatial constraints while learning how to explore them efficiently.

When obstacles are dynamic, the spatial information alone is no longer sufficient. The UAV must also understand how the environment is changing over time. A moving obstacle that is currently far away may soon cross the UAV's path, while a nearby obstacle moving away may pose little risk. Consequently, the design emphasis shifts from purely spatial perception toward temporal reasoning and collision anticipation. Current observations are often supplemented with information from previous time steps, such as changes in sensor measurements, historical observations, estimated obstacle velocities, or short-term memory. These additions allow the policy to distinguish between different motion patterns and react before a collision becomes unavoidable. Learning strategies also become more safety-oriented because near-collision and rapidly changing situations are especially informative. Experience replay methods can give greater attention to difficult or safety-critical interactions, while reward functions can encourage the UAV to maintain safe separation not only from the current obstacle position but also from potential future collision regions. Some designs further combine learned policies with explicit safety guidance so that clearly unsafe actions can be corrected. In this setting, the central requirement is no longer simply to identify free space, but to understand how free space may change and to act early enough to remain safe.

Wind disturbance introduces another form of complexity because uncertainty affects the UAV itself rather than only the surrounding environment. Even when the desired path is known and obstacles are correctly perceived, wind can push the vehicle away from its intended trajectory, reduce tracking accuracy and change the effectiveness of control actions. The DRL system must therefore learn not only where to move, but also how to remain stable and effective when the relationship between action and motion changes. This shifts the design emphasis toward robustness, adaptation, and disturbance compensation. At the low-level control layer, DRL is often combined with classical control or disturbance estimation mechanisms so that learned policies can compensate for persistent or changing external forces. Robust training can expose the policy to different disturbance conditions, allowing it to maintain performance when wind strength or direction changes. At the navigation level, adaptation becomes important because wind can alter the safety, cost, or feasibility of candidate routes. Memory, adaptive policies, and rapid replanning can therefore help the UAV respond to changing disturbance patterns rather than following a fixed strategy.

These observations suggest that effective DRL navigation depends less on selecting a universally superior algorithm and more on adapting the overall learning framework to the challenges of the operating environment. The same base algorithm can perform very differently depending on the detail of its design. As environmental complexity increases, these supporting design choices become increasingly important because the policy must learn from richer information while remaining reliable under conditions that may be difficult to encounter sufficiently during training. Therefore, DRL design for complex UAV navigation should be viewed as a coordinated combination of perception, state representation, learning strategy, and control rather than as a simple choice among individual reinforcement learning algorithms. This perspective also explains why recent research increasingly modifies or augments established DRL methods instead of relying solely on improvements to the base algorithm itself.

4. UAV simulation and performance metrics

4.1. UAV simulators

Drone simulation plays two crucial roles in DRL-based navigation: (i) low-cost training that enables the extensive trial-and-error required by reinforcement learning without risking hardware, safety, or regulatory compliance; (ii) Test and evaluate new algorithms, sensors, and autonomy stacks under controlled, repeatable experiments. It supports environments for quick prototyping before committing to field tests. In this section, we focus on a set of widely used, readily accessible UAV simulators that are commonly adopted in DRL workflows on mainstream operating systems: *AirSim* [137], *Flightmare* [138], *gym-pybullet-drones* [139], and *Gazebo* [140], while noting that many other platforms exist (e.g., proprietary or domain-specific simulators) and may be appropriate depending on different application requirements. A full comparison of the discussed simulators is presented in Table 6.

AirSim is an open-source, cross-platform simulator built upon the Unreal Engine [141] and was created by Microsoft Research as a platform for AI experimentation [137]. Its primary strengths for DRL navigation are **high-fidelity visual sensing** and a practical autonomy workflow. The simulator exposes Remote Procedure Call (RPC) APIs for vehicle control and data retrieval with client support languages including C++, Python, C#, and Java and supports Software In The Loop (SITL) with PX4/ArduPilot as well as Hardware In The Loop (HITL) with PX4, which makes it suitable for perception-driven navigation and sim-to-real validation [137]. In addition, the simulator supports widely used robotic platforms, such as the Robot Operating System (ROS) [142]. The main weaknesses are the computational cost associated with photorealistic simulation and the engineering effort required to align simulated dynamics and sensors with target hardware. Additionally, AirSim's maintenance trajectory should be monitored because Microsoft has announced plans to archive the original AirSim repository which may affect long-term reproducibility and tooling stability. Notable Airsim usecases include vision-based collision avoidance, learning-based navigation in complex urban-like scenes, and dataset generation using its camera and sensor APIs.

Flightmare explicitly targets the speed–fidelity dilemma by decoupling a configurable Unity-based rendering engine [143] from a separate physics engine for dynamics simulation [138]. This design enables extremely high simulation throughput. It reported up to 230 Hz rendering and up to 200,000 Hz physics on a laptop, supporting large-scale parallel DRL training for hundreds of quadrotors. Its strengths include

scalable training, multi-modal sensors (e.g., RGB/depth/segmentation) and interfaces for extracting 3D scene information (e.g., point clouds), which are valuable for learning in cluttered 3D environments [138]. The main limitations are that high-throughput physics and photo-realistic rendering do not automatically guarantee real-world correspondence. While Flightmare can interface with robotics middleware via ROS wrappers, careful calibration and sim-to-real transfer strategies are still required. The simulator is typically positioned as a fast, experiment/evaluation platform in research rather than an end-to-end integration with full autopilot stacks (e.g. PX4/ArduPilot) and HITL testing. In the original demonstrations, Flightmare was used for learning quadrotor control policies and for collision-free planning in complex 3D scenes [138].

gym-pybullet-drones represents a lightweight UAV simulator built on PyBullet/Bullet physics [144] with a standard Gym-style interface, designed explicitly for single and multi-agent learning and control research [139]. A key advantage is its tight alignment with common RL tooling: the standard environment interface simplifies integration with libraries such as Stable-Baselines3 or RLlib, and headless execution supports efficient parallel training for sample-hungry algorithms. It also provides physics-based features such as realistic collisions and aerodynamic effects that make it credible for control-centric evaluation [139]. However, its strengths come with limitations. PyBullet-based simulations typically emphasize rigid-body dynamics and offer limited capability for photorealistic rendering, sensor fidelity, and rich world modeling. Currently the main repository does not have native ROS integration in the core package, which may require users to build custom bridges when testing ROS-based autonomy stacks. As a result, the simulator can be viewed as a fast, credible, and controllable experimental platform for UAV control research in single and multi-agent tasks such as hover stabilization, trajectory tracking, and formation flight.

Gazebo is a widely used open-source robotics simulator that supports UAV research by combining physics-based dynamics, extensible sensor simulation and system integration via plugins and messaging interfaces, and it is widely used for closed-loop testing of full autonomy stacks in ROS-centric ecosystems [140]. A major advantage is its ecosystem compatibility: researchers can connect Gazebo to common robotics middleware (e.g. ROS), and to popular autopilot stacks (e.g., PX4 or ArduPilot) for SITL evaluation. The plugin architecture allows custom models for vehicles, sensors, wind disturbances, and environment interactions, making Gazebo suitable for system-level experiments that span perception, planning, and control. The main drawbacks for DRL are typically lower training throughput than minimalist RL environments and non-trivial configuration overhead when scaling scenario diversity, sensors, and multi-vehicle setups [145]. Overall, Gazebo is best suited for researchers who need an open, modular, and robotics-native simulator to validate autonomy pipelines and autopilot integration. When photorealistic perception and deployment robustness are primary goals, it is often complemented with higher fidelity rendering engines or real-world testing.

4.2. Performance metrics

Selecting appropriate evaluation metrics for autonomous UAV navigation is essential for producing fair comparisons across algorithms and for deriving useful conclusions about performance under realistic operational constraints. Because navigation policies are often trained and tested in task-specific simulators or environments, a poorly chosen metric set can yield over-optimistic results or an incomplete assessment (e.g., reporting goal attainment without characterizing failure modes). Therefore, we recommend a robust evaluation protocol that should cover at least three dimensions: safety, efficiency, and robustness. Table 7 provides a full overview of the discussed performance metrics.

A standard safety or reliability indicator is *Success Rate (SR)*, which is defined as the percentage of episodes/experiments in which the UAV reaches the goal without collision or timeout. It remains the most widely reported metrics particularly in unknown or partially observed environments. However, SR alone is insufficient and it should be complemented by *Crash Rate (CR)*, which is defined as the percentage of episodes/experiments ending in collision typically due to obstacle-avoidance failure. *Lost Rate (LR)* is another important metric which captures the scenarios that UAV neither crashes nor reaches the goal, typically due to timeouts, oscillatory behaviors, or local minima. These failure patterns are frequently observed in large-scale or sparse-reward RL settings. For tasks with an explicit reference trajectory such as corridor following, precision landing, target tracking, *Root Mean Square Error (RMSE)* provides a principled measure of tracking precision, either relative to the desired trajectory or position. Methodologically, RMSE is attractive because squaring the error penalizes occasional large deviations (e.g., overshoot, estimator dropouts, aggressive oscillations), which often correlate with poor stability or weak disturbance rejection.

Beyond binary safety outcomes, efficiency-oriented metrics are necessary to prevent overly favorable evaluations of algorithms that succeed only by taking unnecessarily long or conservative routes in complex environments. *Path Length (PL)*, defined as the total distance traveled typically averaged over successful episodes, is widely used to measure navigational efficiency. When safety constraints are satisfied, shorter trajectories generally imply better planning and fewer energy expenditures. PL is therefore commonly reported alongside SR and CR, and used in cluttered scenes where a policy can achieve a high SR by wandering until it eventually reaches the goal, at the cost of impractical mission times, higher energy consumption and higher exposure to risk. *Success weighted by Path Length (SPL)* is a standard metric that jointly captures task completion and navigation efficiency, originally proposed to avoid reporting success without accounting for how circuitous the agent's motion was [146]. In practice, SPL penalizes inefficient paths while preserving a binary notion of success, which yields a single metric that better reflects operational usefulness than SR alone. Although originated in embodied navigation benchmarks, SPL can be easily extended to UAV navigation when an approximate optimal path can be estimated, such as via global planners, occupancy-grid shortest paths, or best-known trajectories from repeated trials. However, SPL measures distance efficiency, not directly linked to time, energy or UAV dynamics. For instance, A distance-optimal route can be slow if it forces frequent deceleration, tight cornering, stop-and-go motion. In environment with nontrivial dynamics, distance-optimal behavior may not align with time-optimal or energy-optimal behavior.

Learning-based evaluation signals are also widely reported, most notably *Average Reward/Return (AR)*, which is computed as the mean cumulative reward over evaluation episodes. While convenient in (DRL), AR is inherently reward-design dependent: different shaping function returns can make them incomparable across algorithms and can obscure which behavioral properties improved. Consequently, reward-based metrics should be reported primarily as within-study analysis and should be accompanied by aforementioned safety or efficiency metrics. *Average Steps to Goal* and *Time to Goal (TTG)* quantify how quickly an autonomous UAV reaches its objective, commonly in discrete decision steps in RL environments. These metrics are useful because they capture learning speed and efficiency in a directly interpretable way. In RL-based navigation, average steps or time to goal (TTG) provides additional insight into learning progress. As policies improve during training, they tend to reduce unnecessary wandering and dead ends, which is reflected in decreasing TTG values. Plotting TTG (or steps) over training iterations therefore helps assess convergence speed toward an effective policy, which complements reward curves that may be misleading due to reward shaping.

Robustness metrics are also important for evaluating UAV navigation performance under environmental disturbance and uncertainty. Therefore, a typical evaluation should quantify *sensitivity to disturbances* by varying perturbation magnitudes and reporting degradation curves rather than a single-point result. Studies should report how

Table 6
Comparison of common UAV simulators used in RL research.

simulator	Rendering engine	Programming API	ROS integration	Autopilot	Primary Research focus
Airsim	Unreal Engine	Python/C++	ROS wrapper support	Explicitly supports PX4/ArduPilot SITL/HITL	Perception-heavy navigation, synthetic data generation, and SITL or HITL simulation.
Flightmare	Unity	Python/C++, built-in wrappers for gym interface	ROS wrapper support	possible interface with PX4 via bridge	High simulation throughput for quadrotor learning/control with optional photorealistic rendering
gym-pybullet-drones	Pybullet/ OpenGL	Python, native gym-style environment and SB3 workflow	Not supported	Not supported	RL/control benchmarking (single + multi-agent) with fast, lightweight physics and standard Gym API
Gazebo	ODE/ Bullet/ Simbody/ DART	C++	Native ROS-centric ecosystem	Native support for PX4 and ArduPilot	System-level robotics integration (ROS), physics + sensors + plugins; widely used for SITL/HITL stacks

core navigation outcomes (e.g., SR, CR, LR, PL, etc.) change as a function of perturbation level, ideally identifying failure thresholds and safety margins. Such curves are more informative than aggregate averages of these metrics because they characterize operational envelopes and reveal whether performance collapses abruptly or degrades gracefully. In addition, robustness should address *generalization across environments*. A typical approach involves evaluating performance using the same safety or efficiency metrics on unseen maps and conditions that induce distribution shift in geometry and dynamics. Evaluations should therefore include test suites with varying obstacle densities, corridor widths, clutter types, and scale, as well as changes in textures, lighting, weather effects, and sensor noise profiles to probe perception-driven failure modes. For dynamic environments, robustness should also be assessed under different patterns of moving obstacles or agents (e.g., pedestrians, vehicles, other UAVs), including variations in speed, interaction frequency, and behavioral stochasticity.

5. Current challenges and future directions

In this work, we summarize and compare various approaches proposed in the literature for improving mapless UAV autonomous navigation in complex environments using reinforcement learning methods. The environments considered include navigation in cluttered spaces, navigation with dynamic obstacles, and navigation under wind disturbances. Based on the analysis presented in the previous sections, the following paragraphs discuss the current challenges and outline potential future research directions.

A central current challenge for mapless UAV navigation in complex environments is that many DRL systems depend on a multi-stage perception pipeline whose errors directly shape both decision quality and safety. In practice, dynamic obstacle avoidance often requires reliable detection, depth estimation, and tracking. When RGB-D signals degrade due to clutter, poor lighting, or heavy occlusion, the downstream policy may receive incomplete or misleading inputs. This error propagation becomes particularly problematic when the safety mechanism is also driven by the same perception outputs, because a missed detection can remove a critical constraint, while false alarms can trigger unnecessary evasive behavior. As a result, future work should emphasize robustness to imperfect sensing as a first-class objective rather than an implicit assumption. Promising directions include designing policies that can cope gracefully with missing or uncertain observations and evaluating methods under controlled sensing degradations that reflect realistic failure modes instead of only clean sensor conditions.

Secondly, closely related challenge concerns how dynamic agents are represented and anticipated. Many existing designs compress the surrounding traffic into a fixed-capacity encoding, such as the nearest few moving obstacles, which can break down in crowded scenes. The closest agents are not always the most dangerous. For instance, a fast mover slightly farther away, or cross-traffic approaching from the side, may pose greater risk than a nearby but slow obstacle. Moreover, when systems attempt short-term prediction of obstacle motion, performance often deteriorates rapidly as the prediction horizon increase which limits the usefulness of forecasting to near-term reactions. Future work should prioritize scalable representations that can manage variable numbers of moving obstacles without assuming a fixed capacity, along with risk-aware selection mechanisms that rank obstacles by potential threat instead of distance alone. Furthermore, dynamic obstacles should not be treated as isolated observations across time. Incorporating simple temporal patterns such as acceleration and turning can improve the stability of navigation decisions and help prevent delayed or abrupt avoidance maneuvers.

A third challenge concerns how safety is enforced during learning and deployment. Many mapless DRL systems rely on a safety layer that modifies or overrides the policy when a collision seems imminent. Although this approach can reduce crash rates, it often relies on simplified geometric assumptions such as modeling obstacles as spheres and short time-horizon checks. In dense environments, such mechanisms may become overly conservative, resulting in hesitant motion, deadlocks, or unnecessarily long detours as the system repeatedly blocks feasible actions. Furthermore, safety is often encouraged through reward penalties rather than guaranteed by design, which means constraint violations may still occur when the policy encounters unfamiliar conditions or when perception temporarily fails. A key future direction is to move from ad hoc safety fixes toward more principled approaches to constraint handling and transparent safety reporting. This includes explicitly monitoring safety violations beyond collisions, such as near misses, minimum separation distances, and the time spent in unsafe states. It is also important to develop safety mechanisms that remain reliable under sensing uncertainty.

A fourth challenge is control realism, particularly the gap between simplified learning interfaces and realistic control. Many studies restrict motion to a horizontal plane with fixed altitude, use discrete action choices, or assume near-perfect tracking of commanded velocities. These abstractions help learning, but they hide major difficulties that appear in practice such as actuator limits, delays, and the coupling between translation and attitude. Even when a policy outputs continuous acceleration-like commands, the resulting control can fluctuate from step to step, which indicates limited inherent smoothness and raises concerns about stability and energy use. Future work should prioritize learning and evaluation under more realistic control conditions while maintaining overall system practicality. One promising direction is to incorporate fundamental vehicle constraints directly into the learning

Table 7
Common evaluation metrics for learning-based UAV navigation.

Metric	Focus dimension	Typical usecases	Measurement focus	Notable example
Success Rate (SR)	Safety	Point-to-point navigation, obstacle avoidance, exploration with goal	Percentage of episodes/experiments reaching goal without collision or timeout	In most point-to-point UAV navigation studies
Crash Rate (CR)	Safety	Point-to-point navigation, obstacle avoidance	Percentage of episodes/experiment episodes ending in collision	[27], [18], [108], [30]
Lost Rate (LR)	Safety/Efficiency	Point-to-point navigation, large-scale navigation	Percentage of episodes that neither succeed nor crash	[27],[108],[30]
Root Mean Square Error (RMSE)	Safety/Efficiency	Trajectory/target tracking, precision landing	Root Mean Square Error of tracking precision relative to the desired trajectory or position	[132],[64],[19]
Path Length (PL)	Efficiency	Goal-oriented navigation with efficiency considerations	Total distance traveled avg over episodes/experiments	[17],[64],[120]
Success weighted by Path Length (SPL)	Safety/Efficiency	Goal-oriented navigation with efficiency considerations	Success rate weighted by navigation efficiency, computed upon the ratio between the shortest feasible PL and the actual traveled PL	[103],[147],[148]
Average Reward (AR)	Learning-specific	DRL training evaluation	Mean cumulative reward over evaluation episodes	In most RL-based studies
Average Steps to Goal/ Time to Goal (TTG)	Efficiency/ Learning-specific	Time-critical navigation, energy-efficient navigation/DRL training evaluation	Mean steps to reach goal	[120],[118],[92]
Disturbance Sensitivity	Robustness (uncertainty)	Robust navigation control under disturbance	Performance metrics (e.g. SR/CR/LR) vs. perturbation magnitude	[108],[19],[128]
Generalization across environments	Robustness (distribution shift)	Sim-to-real transfer, domain randomization	Performance metrics (e.g. SR/CR/LR) on unseen maps/settings	[27],[17],[18], [107],[134]

problem, such as limits on rapid command changes, speed bounds, and realistic control delays, so that the learned policy does not rely on an idealized simulator.

Lastly, many mapless DRL navigation results remain convincing primarily in simulation or within a narrow set of simulated scenarios that differ only slightly from the training environments. Generalization and transfer to deployments in different environment remain significant challenges. In practice, UAVs must operate under shifts in sensing and dynamics, including different cameras or depth sensors, changing lighting and weather conditions, variations in obstacle geometry and motion that may not be present during training. When such factors are insufficiently represented, the learned policies may fail abruptly and safety mechanisms may also behave unpredictably. This is because their underlying assumptions no longer reflect real-world conditions. Therefore, sim-to-real transfer should be considered a core research objective rather than a peripheral discussion in future research. Potential directions include designing training environments that intentionally vary sensor quality, noise characteristics, and latency. Expanding the diversity of obstacle types and spatial layouts during training can further improve robustness. For evaluation, performance should be assessed in explicitly defined unseen environments instead of relying solely on variations of the training maps.

6. Conclusion

In this survey, we reviewed how deep reinforcement learning is being applied to mapless UAV navigation in complex environments. We first introduced the foundational topics, including UAV types, kinematics, and dynamics, since the maneuvers a UAV can safely perform are ultimately determined by these physical constraints. Building on that foundation, we discussed Deep reinforcement learning, core components of autonomous UAV navigation, including autonomy levels, the differences between map-based and mapless navigation, and the roles of path planning, trajectory planning, and reactive control. The main body of the survey organized recent DRL approaches by the environment challenges they target. In cluttered spaces, many methods show strong progress in local obstacle avoidance, but they often rely on simplified sensing and motion assumptions. With dynamic obstacles, the key difficulty is not only avoiding collisions, but also handling incomplete observations and fast-changing interactions. Under wind disturbance, we saw that robustness is often improved through compensation, adaptation, or training with disturbances, yet many studies still test only a narrow set of wind conditions. Across these topics, a common theme is that impressive results in simulation do not always translate into reliable behavior in real flight, especially when perception degrades, dynamics differ, or safety must be guaranteed.

We also highlighted simulators and performance metrics, since evaluation quality often determines how much we can trust a claim. Future work should place more weight on realistic sensing, stronger safety reporting beyond collision counts, smoother and more stable control outputs, and clearer generalization tests across environments, sensors, and disturbances. Overall, mapless DRL is a promising direction for UAV autonomy, but the next step is to make these systems easier to trust, compare, and deploy in the complex settings they aim to serve.

CRediT authorship contribution statement

Dawei Ni: Writing – review & editing, Writing – original draft, Investigation, Data curation, Conceptualization. **Steven H.H. Ding:** Writing – review & editing, Validation, Supervision, Conceptualization. **Benjamin C.M. Fung:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is supported by NSERC CREATE Grants (CREATE 596346-2025) and Canada Research Chairs Program (CRC-2019-00041).

Data availability

No data was used for the research described in the article.

References

[1] Syed Agha Hassnain Mohsan, Nawaf Qasem Hamood Othman, Yanlong Li, Mohammed H. Alsharif, Muhammad Asghar Khan, Unmanned aerial vehicles (UAVs): Practical aspects, applications, open challenges, security issues, and future trends, Intell. Serv. Robot. 16 (1) (2023) 109–137.
[2] Shumaila Javaid, Nasir Saeed, Zakria Qadir, Hamza Fahim, Bin He, Houbing Song, Muhammad Bilal, Communication and control in collaborative UAVs: Recent advances and future trends, IEEE Trans. Intell. Transp. Syst. 24 (6) (2023) 5719–5739.
[3] Sifat Rezwan, Wooyeol Choi, Artificial intelligence approaches for UAV navigation: Recent advances and future challenges, IEEE Access 10 (2022) 26320–26339.
[4] Yuncheng Lu, Zhucun Xue, Gui-Song Xia, Liangpei Zhang, A survey on vision-based UAV navigation, Geo-Spatial Inf. Sci. 21 (1) (2018) 21–32.
[5] Yifang Ma, Zhenyu Wang, Hong Yang, Lin Yang, Artificial intelligence applications in the development of autonomous vehicles: A survey, IEEE/CAA J. Autom. Sin. 7 (2) (2020) 315–329.
[6] David P. Watson, David H. Scheidt, Autonomous systems, Johns Hopkins APL Tech. Dig. 26 (4) (2005) 368–376.
[7] Taha Elmokadem, Andrey V. Savkin, Towards fully autonomous UAVs: A survey, Sensors 21 (18) (2021) 6223.
[8] František Duchoň, Andrej Babinec, Martin Kajan, Peter Beňo, Martin Florek, Tomáš Fico, Ladislav Jurišica, Path planning with modified a star algorithm for a mobile robot, Procedia Eng. 96 (2014) 59–69.
[9] Kwangjin Yang, Seng Keat Gan, Salah Sukkarieh, A Gaussian process-based RRT planner for the exploration of an unknown and cluttered environment with a UAV, Adv. Robot. 27 (6) (2013) 431–443.
[10] Yong-bo Chen, Guan-chen Luo, Yue-song Mei, Jian-qiao Yu, Xiao-long Su, UAV path planning using artificial potential field method updated by optimal control theory, Int. J. Syst. Sci. 47 (6) (2016) 1407–1420.
[11] Saeid Sedighi, Duong-Van Nguyen, Klaus-Dieter Kuhnert, Guided hybrid A-star path planning algorithm for valet parking applications, in: 2019 5th International Conference on Control, Automation and Robotics, ICCAR, IEEE, 2019, pp. 570–575.
[12] Abdurrahim Sonmez, Emre Kocyigit, Emin Kugu, Optimal path planning for UAVs using genetic algorithm, in: 2015 International Conference on Unmanned Aircraft Systems, ICUAS, IEEE, 2015, pp. 50–55.
[13] Pakorn Poksawat, Liuping Wang, Abdulghani Mohamed, Gain scheduled attitude control of fixed-wing UAV with automatic controller tuning, IEEE Trans. Control Syst. Technol. 26 (4) (2017) 1192–1203.
[14] En-Hui Zheng, Jing-Jing Xiong, Ji-Liang Luo, Second order sliding mode control for a quadrotor UAV, ISA Trans. 53 (4) (2014) 1350–1356.
[15] Heriberto Ramirez-Rodriguez, Vicente Parra-Vega, Anand Sanchez-Orta, Octavio Garcia-Salazar, Robust backstepping control based on integral sliding modes for tracking of quadrotors, J. Intell. Robot. Syst. 73 (1) (2014) 51–66.
[16] Bo Zhao, Bin Xian, Yao Zhang, Xu Zhang, Nonlinear robust adaptive tracking control of a quadrotor UAV via immersion and invariance methodology, IEEE Trans. Ind. Electron. 62 (5) (2014) 2891–2902.
[17] GUO Tong, Nan Jiang, LI Biyue, ZHU Xi, WANG Ya, DU Wenbo, UAV navigation in high dynamic environments: A deep reinforcement learning approach, Chin. J. Aeronaut. 34 (2) (2021) 479–489.
[18] Chao Wang, Jian Wang, Yuan Shen, Xudong Zhang, Autonomous navigation of UAVs in large-scale complex environments: A deep reinforcement learning approach, IEEE Trans. Veh. Technol. 68 (3) (2019) 2124–2136.
[19] Bodi Ma, Zhenbao Liu, Qingqing Dang, Wen Zhao, Jingyan Wang, Yao Cheng, Zhirong Yuan, Deep reinforcement learning of UAV tracking control under wind disturbances environments, IEEE Trans. Instrum. Meas. 72 (2023) 1–13.
[20] Punarjay Chakravarty, Klaas Kelchtermans, Tom Roussel, Stijn Wellens, Tinne Tuytelaars, Luc Van Eycken, CNN-based single image obstacle avoidance on a quadrotor, in: 2017 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2017, pp. 6369–6374.
[21] Alexandros Kouris, Christos-Savvas Bouganis, Learning to fly by myself: A self-supervised cnn-based approach for autonomous navigation, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2018, pp. 1–9.
[22] Xi Dai, Yuxin Mao, Tianpeng Huang, Na Qin, Deqing Huang, Yanan Li, Automatic obstacle avoidance of quadrotor UAV via CNN-based learning, Neurocomputing 402 (2020) 346–358.
[23] Sunggoo Jung, Sunyou Hwang, Heemin Shin, David Hyunchul Shim, Perception, guidance, and navigation for indoor autonomous drone racing using deep learning, IEEE Robot. Autom. Lett. 3 (3) (2018) 2539–2544.
[24] Yuxi Li, Deep reinforcement learning: An overview, 2017, arXiv preprint arXiv: 1701.07274.
[25] Richard S. Sutton, Andrew G. Barto, et al., Reinforcement Learning: An Introduction, volume 1, MIT press Cambridge, 1998.
[26] Jiaping Xiao, Rangya Zhang, Yuhang Zhang, Mir Feroskhan, Vision-based learning for drones: A survey, IEEE Trans. Neural Networks Learn. Syst. (2025).
[27] Sitong Zhang, Yibing Li, Qianhui Dong, Autonomous navigation of UAV in multi-obstacle environments based on a deep reinforcement learning approach, Appl. Soft Comput. 115 (2022) 108194.
[28] Fadi AlMahamid, Katarina Grolinger, Autonomous unmanned aerial vehicle navigation using reinforcement learning: A systematic review, Eng. Appl. Artif. Intell. 115 (2022) 105321.
[29] Abhik Singla, Sindhu Padakandla, Shalabh Bhatnagar, Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge, IEEE Trans. Intell. Transp. Syst. 22 (1) (2019) 107–118.
[30] Yongfeng Yin, Zhetao Wang, Lili Zheng, Qingran Su, Yang Guo, Autonomous UAV navigation with adaptive control based on deep reinforcement learning, Electronics 13 (13) (2024) 2432.
[31] Faiyaz Ahmed, JC Mohanta, Anupam Keshari, Pankaj Singh Yadav, Recent advances in unmanned aerial vehicles: a review, Arab. J. Sci. Eng. 47 (7) (2022) 7963–7984.
[32] Yu Bai, Hui Zhao, Xin Zhang, Zheng Chang, Riku Jäntti, Kun Yang, Toward autonomous multi-UAV wireless network: A survey of reinforcement learning-based approaches, IEEE Commun. Surv. Tutor. 25 (4) (2023) 3038–3067.
[33] Ahmad Taher Azar, Anis Koubaa, Nada Ali Mohamed, Habiba A. Ibrahim, Zahra Fathy Ibrahim, Muhammad Kazim, Adel Ammar, Bilel Benjdira, Alaa M. Khamis, Ibrahim A. Hameed, et al., Drone deep reinforcement learning: A review, Electronics 10 (9) (2021) 999.
[34] Hengsheng Chen, Yuanguo Lin, Mingjian Fu, Lina Yao, Michael Sheng, A survey on reinforcement learning methods for UAV systems, ACM Comput. Surv. 58 (4) (2025) 1–37.
[35] Leonardo A. Fagundes-Junior, Kevin B. de Carvalho, Ricardo S. Ferreira, Alexandre S. Brandão, Machine learning for unmanned aerial vehicles navigation: An overview, SN Comput. Sci. 5 (2) (2024) 256.
[36] Zongyu Zuo, Cunjia Liu, Qing-Long Han, Jiawei Song, Unmanned aerial vehicles: Control methods and future challenges, IEEE/CAA J. Autom. Sin. 9 (4) (2022) 601–614.
[37] Adnan Khalid, Zohaib Mushtaq, Saad Arif, Kamran Zeb, Muhammad Attique Khan, Sambit Bakshi, Control schemes for quadrotor UAV: taxonomy and survey, ACM Comput. Surv. 56 (5) (2023) 1–32.
[38] Shubhani Aggarwal, Neeraj Kumar, Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges, Comput. Commun. 149 (2020) 270–299.
[39] Michael Jones, Soufiene Djahel, Kristopher Welsh, Path-planning for unmanned aerial vehicles with environment complexity considerations: A survey, ACM Comput. Surv. 55 (11) (2023) 1–39.
[40] Muhammad Yeasir Arafat, Muhammad Morshed Alam, Sangman Moh, Vision-based navigation techniques for unmanned aerial vehicles: Review and challenges, Drones 7 (2) (2023) 89.
[41] Foisal Ahmed, Maksim Jenihhin, A survey on UAV computing platforms: A hardware reliability perspective, Sensors 22 (16) (2022) 6286.
[42] Evanthia Sdoukou, Athanasios Milidonis, Konstantinos Efstathiou, Ioannis Voyiatzis, A survey of UAV hardware selection, J. Eng. Appl. Sci. 72 (1) (2025) 88.
[43] Hazim Shakhatreh, Ahmad H. Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, Mohsen Guizani, Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges, IEEE Access 7 (2019) 48572–48634.
[44] Kimon P. Valavanis, George J. Vachtsevanos, Handbook of unmanned aerial vehicles, Springer Publishing Company, Incorporated, 2014.
[45] Randal W. Beard, Timothy W. McLain, Small Unmanned Aircraft: Theory and Practice, Princeton University Press, 2012.
[46] Robert Mahony, Vijay Kumar, Peter Corke, Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor, IEEE Robot. Autom. Mag. 19 (3) (2012) 20–32.
[47] Teppo Luukkonen, Modelling and control of quadcopter, Indep. Res. Proj. Appl. Math. Espoo 22 (22) (2011) 1–24.
[48] Julius A. Marshall, Wei Sun, Andrea L'Afflitto, A survey of guidance, navigation, and control systems for autonomous multi-rotor small unmanned aerial systems, Annu. Rev. Control. 52 (2021) 390–427.

[49] Anthony R. Cassandra, A survey of POMDP applications, in: Working Notes of AAAI 1998 Fall Symposium on Planning with Partially Observable Markov Decision Processes, vol. 1724, 1998.

[50] Ivo Grondman, Lucian Busoniu, Gabriel A.D. Lopes, Robert Babuska, A survey of actor-critic reinforcement learning: Standard and natural policy gradients, IEEE Trans. Syst. Man, Cybern. Part C (Applications Reviews) 42 (6) (2012) 1291–1307.

[51] Liyuan Zheng, Lillian Ratliff, Constrained upper confidence reinforcement learning, in: Learning for Dynamics and Control, PMLR, 2020, pp. 620–629.

[52] Yali Du, Lei Han, Meng Fang, Ji Liu, Tianhong Dai, Dacheng Tao, Liir: Learning individual intrinsic reward in multi-agent reinforcement learning, Adv. Neural Inf. Process. Syst. 32 (2019).

[53] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller, Playing atari with deep reinforcement learning, 2013, arXiv preprint arXiv:1312.5602.

[54] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.

[55] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, Daan Wierstra, Continuous control with deep reinforcement learning, 2015, arXiv preprint arXiv:1509.02971.

[56] Scott Fujimoto, Herke Hoof, David Meger, Addressing function approximation error in actor-critic methods, in: International Conference on Machine Learning, PMLR, 2018, pp. 1587–1596.

[57] Bram Bakker, Reinforcement learning with long short-term memory, Adv. Neural Inf. Process. Syst. 14 (2001).

[58] Marc G. Bellemare, Will Dabney, Rémi Munos, A distributional perspective on reinforcement learning, in: International Conference on Machine Learning, PMLR, 2017, pp. 449–458.

[59] Hui-Min Huang, Autonomy levels for unmanned systems (ALFUS) framework: safety and application issues, in: Proceedings of the 2007 Workshop on Performance Metrics for Intelligent Systems, 2007, pp. 48–53.

[60] Emad Ebeid, Martin Skriver, Kristian Husum Terkildsen, Kjeld Jensen, Ulrik Pagh Schultz, A. survey of open-source UAV flight controllers and flight simulators, Microprocess. Microsyst. 61 (2018) 11–20.

[61] Yavinaash Naidu Saravanakumar, Mohamed Thariq Hameed Sultan, Farah Syazwani Shahar, Wojciech Giernacki, Andrzej Łukaszewicz, Marek Nowakowski, Andriy Holovatyy, Sławomir Stępień, Power sources for unmanned aerial vehicles: a state-of-the art, Appl. Sci. 13 (21) (2023) 11932.

[62] Oualid Doukhi, Deok Jin Lee, Deep reinforcement learning for autonomous map-less navigation of a flying robot, IEEE Access 10 (2022) 82964–82976.

[63] Enrico Marchesini, Alessandro Farinelli, Discrete deep reinforcement learning for mapless navigation, in: 2020 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2020, pp. 10688–10694.

[64] Chen-Huan Pi, Wei-Yuan Ye, Stone Cheng, Robust quadrotor control through reinforcement learning with disturbance compensation, Appl. Sci. 11 (7) (2021) 3257.

[65] Halil Ibrahim Ugurlu, Xuan Huy Pham, Erdal Kayacan, Sim-to-real deep reinforcement learning for safe end-to-end planning of aerial robots, Robotics 11 (5) (2022) 109.

[66] Fei Yan, Yi-Sha Liu, Ji-Zhong Xiao, Path planning in complex 3D environments using a probabilistic roadmap method, Int. J. Autom. Comput. 10 (6) (2013) 525–533.

[67] Vladyslav Usenko, Lukas Von Stumberg, Andrej Pangercic, Daniel Cremers, Real-time trajectory replanning for MAVs using uniform B-splines and a 3D circular buffer, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2017, pp. 215–222.

[68] Mina Kamel, Thomas Stastny, Kostas Alexis, Roland Siegwart, Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system, in: Robot Operating System (ROS) the Complete Reference (Volume 2), Springer, 2017, pp. 3–39.

[69] Xiong Bai, Haikun Jiang, Junjie Cui, Kuan Lu, Pengyun Chen, Ming Zhang, UAV path planning based on improved A and DWA algorithms, Int. J. Aerosp. Eng. 2021 (1) (2021) 4511252.

[70] Yong He, Ticheng Hou, Mingran Wang, A new method for unmanned aerial vehicle path planning in complex environments, Sci. Rep. 14 (1) (2024) 9257.

[71] Jonathan D. Gammell, Siddhartha S. Srinivasa, Timothy D. Barfoot, Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 2997–3004.

[72] Bowen Li, Zhenyu Na, Bin Lin, UAV trajectory planning from a comprehensive energy efficiency perspective in harsh environments, IEEE Netw. 36 (4) (2022) 62–68.

[73] Xiaoxue Zhang, Jun Ma, Zilong Cheng, Sunan Huang, Shuzhi Sam Ge, Tong Heng Lee, Trajectory generation by chance-constrained nonlinear MPC with probabilistic prediction, IEEE Trans. Cybern. 51 (7) (2020) 3616–3629.

[74] Geeta Sharma, Sanjeev Jain, Radhe Shyam Sharma, Path planning for fully autonomous uavs-a taxonomic review and future perspectives, IEEE Access (2025).

[75] Stéphane Ross, Narek Melik-Barkhudarov, Kumar Shaurya Shankar, Andreas Wendel, Debadeepta Dey, J. Andrew Bagnell, Martial Hebert, Learning monocular reactive uav control in cluttered natural environments, in: 2013 IEEE International Conference on Robotics and Automation, IEEE, 2013, pp. 1765–1772.

[76] Björn Lindqvist, Sina Sharif Mansouri, Jakub Haluška, George Nikolakopoulos, Reactive navigation of an unmanned aerial vehicle with perception-based obstacle avoidance constraints, IEEE Trans. Control Syst. Technol. 30 (5) (2021) 1847–1862.

[77] Endri Bregu, Nicola Casamassima, Daniel Cantoni, Luca Mottola, Kamin Whitehouse, Reactive control of autonomous drones, in: Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services, 2016, pp. 207–219.

[78] Junseok Lee, Xiangyu Wu, Seung Jae Lee, Mark W. Mueller, Autonomous flight through cluttered outdoor environments using a memoryless planner, in: 2021 International Conference on Unmanned Aircraft Systems, ICUAS, IEEE, 2021, pp. 1131–1138.

[79] Yingxiu Chang, Yongqiang Cheng, Umar Manzoor, John Murray, A review of UAV autonomous navigation in GPS-denied environments, Robot. Auton. Syst. 170 (2023) 104533.

[80] Xiaoyu Ye, Fujun Song, Zongyu Zhang, Qinghua Zeng, A review of small UAV navigation system based on multisource sensor fusion, IEEE Sensors J. 23 (17) (2023) 18926–18948.

[81] Mohannad Alhafnawi, Haythem A. Bany Salameh, Ala'eddin Masadeh, Haitham Al-Obiedollah, Moussa Ayyash, Reyad El-Khazali, Hany Elgala, A survey of indoor and outdoor uav-based target tracking systems: Current status, challenges, technologies, and future directions, IEEE Access 11 (2023) 68324–68339.

[82] Ram Prasad Padhy, Sachin Verma, Shahzad Ahmad, Suman Kumar Choudhury, Pankaj Kumar Sa, Deep neural network for autonomous UAV navigation in indoor corridor environments, Procedia Comput. Sci. 133 (2018) 643–650.

[83] Teodor Tomic, Korbinian Schmid, Philipp Lutz, Andreas Domel, Michael Kassecker, Elmar Mair, Iris Lynne Grixa, Felix Ruess, Michael Suppa, Darius Burschka, Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue, IEEE Robot. Autom. Mag. 19 (3) (2012) 46–56.

[84] Sinahi Dionisio-Ortega, L. Oyuki Rojas-Perez, Jose Martinez-Carranza, Israel Cruz-Vega, A deep learning approach towards autonomous flight in forest environments, in: 2018 International Conference on Electronics, Communications and Computers, CONIELECOMP, IEEE, 2018, pp. 139–144.

[85] Lijuan Zhang, Jiabin Peng, Weiguo Yi, Hang Lin, Lei Lei, Xiaoqin Song, A state-decomposition DDPG algorithm for UAV autonomous navigation in 3-D complex environments, IEEE Internet Things J. 11 (6) (2023) 10778–10790.

[86] Runjia Wu, Fangqing Gu, Hai-lin Liu, Hongjian Shi, UAV path planning based on multicritic-delayed deep deterministic policy gradient, Wirel. Commun. Mob. Comput. 2022 (1) (2022) 9017079.

[87] Omar Bouhamed, Hakim Ghazzai, Hichem Besbes, Yehia Massoud, Autonomous UAV navigation: A DDPG-based deep reinforcement learning approach, in: 2020 IEEE International Symposium on Circuits and Systems, ISCAS, IEEE, 2020, pp. 1–5.

[88] Junqiao Wang, Zhongliang Yu, Dong Zhou, Jiaqi Shi, Runran Deng, Vision-based deep reinforcement learning of unmanned aerial vehicle autonomous navigation using privileged information, Drones 8 (12) (2024) 782.

[89] Hui Lv, Yadong Chen, Shibo Li, Baolong Zhu, Min Li, Improve exploration in deep reinforcement learning for UAV path planning using state and action entropy, Meas. Sci. Technol. 35 (5) (2024) 056206.

[90] Qiyu Sun, Jinbao Fang, Wei Xing Zheng, Yang Tang, Aggressive quadrotor flight using curiosity-driven reinforcement learning, IEEE Trans. Ind. Electron. 69 (12) (2022) 13838–13848.

[91] L. He, N. Aouf, J.F. Whidborne, B. Song, Deep reinforcement learning based local planner for UAV obstacle avoidance using demonstration data. arxiv 2020, 2020, arXiv preprint arXiv:2008.02521.

[92] Chenchen Fu, Xueyong Xu, Yuntao Zhang, Yan Lyu, Yu Xia, Zining Zhou, Weiwei Wu, Memory-enhanced deep reinforcement learning for UAV navigation in 3D environment, Neural Comput. Appl. 34 (17) (2022) 14599–14607.

[93] Yun Chen, Nuria González-Prelcic, Robert W. Heath, Collision-free UAV navigation with a monocular camera using deep reinforcement learning, in: 2020 IEEE 30th International Workshop on Machine Learning for Signal Processing, MLSP, IEEE, 2020, pp. 1–6.

[94] Chunxue Wu, Bobo Ju, Yan Wu, Xiao Lin, Naixue Xiong, Guangquan Xu, Hongyan Li, Xuefeng Liang, UAV autonomous target search based on deep reinforcement learning in complex disaster scene, IEEE Access 7 (2019) 117227–117245.

[95] Bruna G. Maciel-Pearson, Letizia Marchegiani, Samet Akcay, Amir Atapour-Abarghouei, James Garforth, Toby P. Breckon, Online deep reinforcement learning for autonomous UAV navigation and exploration of outdoor environments, 2019, arXiv preprint arXiv:1912.05684.

[96] Yuhan Xie, Minghao Lu, Rui Peng, Peng Lu, Learning agile flights through narrow gaps with varying angles using onboard sensing, IEEE Robot. Autom. Lett. 8 (9) (2023) 5424–5431.

[97] Zun Liu, Yuanqiang Cao, Jianyong Chen, Jianqiang Li, A hierarchical reinforcement learning algorithm based on attention mechanism for UAV autonomous navigation, IEEE Trans. Intell. Transp. Syst. 24 (11) (2022) 13309–13320.
[98] Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, Davide Scaramuzza, Champion-level drone racing using deep reinforcement learning, Nature 620 (7976) (2023) 982–987.
[99] Christos Chronis, Georgios Anagnostopoulos, Elena Politi, George Dimitrakopoulos, Iraklis Varlamis, Dynamic navigation in unconstrained environments using reinforcement learning algorithms, IEEE Access 11 (2023) 117984–118001.
[100] Chenxi Xiao, Peng Lu, Qizhi He, Flying through a narrow gap using end-to-end deep reinforcement learning augmented with curriculum learning and Sim2Real, IEEE Trans. Neural Netw. Learn. Syst. 34 (5) (2021) 2701–2708.
[101] Victoria J. Hodge, Richard Hawkins, Rob Alexander, Deep reinforcement learning for drone navigation using sensor data, Neural Comput. Appl. 33 (6) (2021) 2015–2033.
[102] Oualid Doukhi, Deok-Jin Lee, Deep reinforcement learning for end-to-end local motion planning of autonomous aerial robots in unknown outdoor environments: Real-time flight experiments, Sensors 21 (7) (2021) 2534.
[103] Steven D. Morad, Roberto Mecca, Rudra P.K. Poudel, Stephan Liwicki, Roberto Cipolla, Embodied visual navigation with automatic curriculum learning in real environments, IEEE Robot. Autom. Lett. 6 (2) (2021) 683–690.
[104] Qiyu Sun, Jiaxin Ji, Jinzhen Mu, Jing Xu, Ljupco Kocarev, Jürgen Kurths, Yang Tang, Vision-based hierarchical reinforcement learning for quadrotor uav navigation, IEEE/ASME Trans. Mechatronics (2025).
[105] Myoung Hoon Lee, Jun Moon, Deep reinforcement learning-based model-free path planning and collision avoidance for UAVs: A soft actor–critic with hindsight experience replay approach, Ict Express 9 (3) (2023) 403–408.
[106] Yoonsu Jang, Jongchan Baek, Soohee Han, Hindsight intermediate targets for mapless navigation with deep reinforcement learning, IEEE Trans. Ind. Electron. 69 (11) (2021) 11816–11825.
[107] HU Zijian, GAO Xiaoguang, WAN Kaifang, ZHAI Yiwei, WANG Qianglong, Relevant experience learning: A deep reinforcement learning method for UAV autonomous motion planning in complex unknown environments, Chin. J. Aeronaut. 34 (12) (2021) 187–204.
[108] Kaifang Wan, Xiaoguang Gao, Zijian Hu, Gaofeng Wu, Robust motion control for UAV in dynamic uncertain environments using deep reinforcement learning, Remote. Sens. 12 (4) (2020) 640.
[109] Xuqiong Luo, Qiyuan Wang, Hongfang Gong, Chao Tang, UAV path planning based on the average TD3 algorithm with prioritized experience replay, IEEE Access 12 (2024) 38017–38029.
[110] Fadi AlMahamid, Katarina Grolinger, Viznav: A modular off-policy deep reinforcement learning framework for vision-based autonomous uav navigation in 3d dynamic environments, Drones 8 (5) (2024) 173.
[111] Zhao Feiyu, Li Dayan, Wang Zhengxu, Mao Jianlin, Wang Niya, Autonomous localized path planning algorithm for UAVs based on TD3 strategy, Sci. Rep. 14 (1) (2024) 763.
[112] Zijian Hu, Xiaoguang Gao, Kaifang Wan, Qianglong Wang, Yiwei Zhai, Asynchronous curriculum experience replay: A deep reinforcement learning approach for UAV autonomous motion control in unknown dynamic environments, IEEE Trans. Veh. Technol. 72 (11) (2023) 13985–14001.
[113] Ying Lu, Xiaodan Wang, Yang Yang, Man Ding, Shaochun Qu, Yanfang Fu, LSTM-DQN-APF path planning algorithm empowered by twins in complex scenarios, Appl. Sci. 15 (8) (2025) 4565.
[114] Tongyao Yang, Fengbao Yang, Dingzhu Li, A new autonomous method of drone path planning based on multiple strategies for avoiding obstacles with high speed and high density, Drones 8 (5) (2024) 205.
[115] WANG Fei, ZHU Xiaoping, Zhou Zhou, TANG Yang, Deep-reinforcement-learning-based UAV autonomous navigation and collision avoidance in unknown environments, Chin. J. Aeronaut. 37 (3) (2024) 237–257.
[116] Yupeng Yang, Kai Zhang, Dahai Liu, Houbing Song, Autonomous UAV navigation in dynamic environments with double deep Q-networks, in: 2020 AIAA/IEEE 39th Digital Avionics Systems Conference, DASC, IEEE, 2020, pp. 1–7.
[117] Ender Cetin, Cristina Barrado, Guillem Muñoz, Miquel Macias, Enric Pastor, Drone navigation and avoidance of obstacles through deep reinforcement learning, in: 2019 IEEE/AIAA 38th Digital Avionics Systems Conference, DASC, IEEE, 2019, pp. 1–7.
[118] Yuanda Wang, Haibo He, Changyin Sun, Learning to navigate through complex dynamic environment with modular deep reinforcement learning, IEEE Trans. Games 10 (4) (2018) 400–412.
[119] Shuyuan Liu, Shufan Zou, Xinghua Chang, Huayong Liu, Laiping Zhang, Xiaogang Deng, Autonomous navigation of UAV in complex environment: a deep reinforcement learning method based on temporal attention, Appl. Intell. 55 (5) (2025) 316.
[120] Yuanyuan Sheng, Huanyu Liu, Junbao Li, Qi Han, UAV autonomous navigation based on deep reinforcement learning in highly dynamic and high-density environments, Drones 8 (9) (2024) 516.
[121] Amudhini P. Kalidas, Christy Jackson Joshua, Abdul Quadir Md, Shakila Basheer, Senthilkumar Mohan, Sapiah Sakri, Deep reinforcement learning for vision-based navigation of UAVs in avoiding stationary and mobile obstacles, Drones 7 (4) (2023) 245.
[122] Xiyu Fan, Minghao Lu, Bowen Xu, Peng Lu, Flying in highly dynamic environments with end-to-end learning approach, IEEE Robot. Autom. Lett. (2025).
[123] Zhefan Xu, Xinming Han, Haoyu Shen, Hanyu Jin, Kenji Shimada, Navrl: Learning safe flight in dynamic environments, IEEE Robot. Autom. Lett. (2025).
[124] Dongliang Wang, Yongqiang Jin, Weicheng Luo, Yijing Yang, Senyi Zhang, Yong Gao, A hierarchical visual navigation algorithm for UAVs integrating artificial potential field and deep reinforcement learning, Sensors 26 (16) (2026) 5196.
[125] Jiandong Liu, Wei Luo, Guoqing Zhang, Ruihao Li, Unmanned aerial vehicle path planning in complex dynamic environments based on deep reinforcement learning, Machines 13 (2) (2025) 162.
[126] Xiaohai Ren, Na Geng, Yong Zhang, Lei Xiao, Dunwei Gong, PG-ITD3: A potential field-guided deep reinforcement learning approach for UAV path planning after disaster, IEEE Trans. Autom. Sci. Eng. (2025).
[127] Zhihui Feng, Xitai Na, Shiji Hai, Qingbin Sun, Jinshuo Shi, Deep reinforcement learning for uav target search and continuous tracking in complex environments with gaussian process regression and prior policy embedding, Electronics 14 (7) (2025) 1330.
[128] Hao-Lun Hsu, Haocheng Meng, Shaocheng Luo, Juncheng Dong, Vahid Tarokh, Miroslav Pajic, Reforma: Robust reinforcement learning via adaptive adversary for drones flying under disturbances, in: 2024 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2024, pp. 5169–5175.
[129] Qun Ma, Yibo Wu, Muhammad Usman Shoukat, Yukai Yan, Jun Wang, Long Yang, Fuwu Yan, Lirong Yan, Deep reinforcement learning-based wind disturbance rejection control strategy for uav, Drones 8 (11) (2024) 632.
[130] Yahui Ren, Feng Zhu, Shuaishuai Sui, Zhengming Yi, Kai Chen, Enhancing quadrotor control robustness with multi-proportional-integral-derivative self-attention-guided deep reinforcement learning, Drones 8 (7) (2024) 315.
[131] Qiuyu Yu, Lingkun Luo, Bing Liu, Shiqiang Hu, Re-planning of quadrotors under disturbance based on meta reinforcement learning, J. Intell. Robot. Syst. 107 (1) (2023) 13.
[132] Bodi Ma, Zhenbao Liu, Wen Zhao, Jinbiao Yuan, Hao Long, Xiao Wang, Zhirong Yuan, Target tracking control of UAV through deep reinforcement learning, IEEE Trans. Intell. Transp. Syst. 24 (6) (2023) 5983–6000.
[133] Aditya M. Deshpande, Ali A. Minai, Manish Kumar, Robust deep reinforcement learning for quadcopter control, IFAC-PapersOnLine 54 (20) (2021) 90–95.
[134] Yuanda Wang, Jia Sun, Haibo He, Changyin Sun, Deterministic policy gradient with integral compensator for robust quadrotor control, IEEE Trans. Syst. Man, Cybern.: Syst. 50 (10) (2019) 3713–3725.
[135] Jiahao Wu, Yang Ye, Jing Du, Multi-objective reinforcement learning for autonomous drone navigation in urban areas with wind zones, Autom. Constr. 158 (2024) 105253.
[136] Ricardo Bedin Grando, Junior Costa de Jesus, Victor Augusto Kich, Alisson Henrique Kolling, Paulo Lilles Jorge Drews-Jr, Double critic deep reinforcement learning for mapless 3d navigation of unmanned aerial vehicles, J. Intell. Robot. Syst. 104 (2) (2022) 29.
[137] Shital Shah, Debadeepta Dey, Chris Lovett, Ashish Kapoor, Airsim: High-fidelity visual and physical simulation for autonomous vehicles, in: Field and Service Robotics: Results of the 11th International Conference, Springer, 2017, pp. 621–635.
[138] Yunlong Song, Selim Naji, Elia Kaufmann, Antonio Loquercio, Davide Scaramuzza, Flightmare: A flexible quadrotor simulator, in: Conference on Robot Learning, PMLR, 2021, pp. 1147–1157.
[139] Jacopo Panerati, Hehui Zheng, SiQi Zhou, James Xu, Amanda Prorok, Angela P. Schoellig, Learning to fly—a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2021, pp. 7512–7519.
[140] Nathan Koenig, Andrew Howard, Design and use paradigms for gazebo, an open-source multi-robot simulator, in: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566), 3, Ieee, 2004, pp. 2149–2154.
[141] Andrew Sanders, An Introduction to Unreal Engine 4, AK Peters/CRC Press, 2016.
[142] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y. Ng, et al., ROS: an open-source robot operating system, in: ICRA Workshop on Open Source Software, volume 3, Kobe, 2009, p. 5.
[143] John K. Haas, A History of the Unity Game Engine, Worcester, MA, USA, 2014.
[144] Hantao He, Junxing Zheng, Quan Sun, Zhaochao Li, Simulation of realistic particles with bullet physics engine, in: E3S Web of Conferences, volume 92, EDP Sciences, 2019, p. 14004.
[145] Fadri Furrer, Michael Burri, Markus Achtelik, Roland Siegwart, Rotors—a modular gazebo mav simulator framework, in: Robot Operating System (ROS) the Complete Reference (Volume 1), Springer, 2016, pp. 595–625.
[146] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al., On evaluation of embodied navigation agents, 2018, arXiv preprint arXiv:1807.06757.

[147] Bardienus P. Duisterhof, Srivatsan Krishnan, Jonathan J. Cruz, Colby R. Banbury, William Fu, Aleksandra Faust, Guido CHE de Croon, Vijay Janapa Reddi, Learning to seek: Autonomous source seeking with deep reinforcement learning onboard a nano drone microcontroller, 2019, arXiv preprint arXiv:1909.11236.

[148] Xiangyu Wang, Donglin Yang, Ziqin Wang, Hohin Kwan, Jinyu Chen, Wenjun Wu, Hongsheng Li, Yue Liao, Si Liu, Towards realistic uav vision-language navigation: Platform, benchmark, and methodology, 2024, arXiv preprint arXiv:2410.07087.