electronics

MDPI

Review

Drone Deep Reinforcement Learning: A Review

Ahmad Taher Azar [1,2,*], Anis Koubaa [1], Nada Ali Mohamed [3], Habiba A. Ibrahim [4], Zahra Fathy Ibrahim [3], Muhammad Kazim [1,5], Adel Ammar [1], Bilel Benjdira [1], Alaa M. Khamis [6], Ibrahim A. Hameed [7] and Gabriella Casalino [8]

1 College of Computer & Information Sciences, Prince Sultan University, Riyadh 11586, Saudi Arabia; akoubaa@psu.edu.sa (A.K.); mkazim@psu.edu.sa (M.K.); aammar@psu.edu.sa (A.A.); bbenjdira@psu.edu.sa (B.B.)
2 Faculty of Computers and Artificial Intelligence, Benha University, Benha 13518, Egypt
3 School of Engineering and Applied Sciences, Nile University Campus, Sheikh Zayed District, Juhayna Square, 6th of October City, Giza 60411, Egypt; N.Ali@nu.edu.eg (N.A.M.); Z.Fathy@nu.edu.eg (Z.F.I.)
4 Smart Engineering Systems Research Center (SESC), Nile University, Sheikh Zayed City, Giza 12588, Egypt; h.ibrahim@nu.edu.eg
5 Research Institute of Intelligent Control and Systems, Harbin Institute of Technology, Harbin 150080, China;
6 General Motors Canada, 500 Wentworth St W, Oshawa ON L1J 6J2, Canada; alaakhamis@gmail.com
7 Department of ICT and Natural Sciences, Norwegian University of Science and Technology, Larsgårdsvegen, 2, 6009 Ålesund, Norway; ibib@ntnu.no
8 Department of Informatics, University of Bari, 70125 Bari, Italy; gabriella.casalino@uniba.it
* Correspondence: aazar@psu.edu.sa or ahmad.azar@fci.bu.edu.eg

Abstract: Unmanned Aerial Vehicles (UAVs) are increasingly being used in many challenging and diversified applications. These applications belong to the civilian and the military fields. To name a few; infrastructure inspection, traffic patrolling, remote sensing, mapping, surveillance, rescuing humans and animals, environment monitoring, and Intelligence, Surveillance, Target Acquisition, and Reconnaissance (ISTAR) operations. However, the use of UAVs in these applications needs a substantial level of autonomy. In other words, UAVs should have the ability to accomplish planned missions in unexpected situations without requiring human intervention. To ensure this level of autonomy, many artificial intelligence algorithms were designed. These algorithms targeted the guidance, navigation, and control (GNC) of UAVs. In this paper, we described the state of the art of one subset of these algorithms: the deep reinforcement learning (DRL) techniques. We made a detailed description of them, and we deduced the current limitations in this area. We noted that most of these DRL methods were designed to ensure stable and smooth UAV navigation by training computer-simulated environments. We realized that further research efforts are needed to address the challenges that restrain their deployment in real-life scenarios.

Keywords: unmanned aerial vehicles; UAVs; guidance; navigation; control; machine learning; deep reinforcement learning (DRL); literature review

Citation: Azar, A.T.; Koubaa, A.; Ali Mohamed, N.; Ibrahim, H.A.; Ibrahim, Z.F.; Kazim, M.; Ammar, A.; Benjdira, B.; Khamis, A.; Hameed, I.A.; et al. Drone Deep Reinforcement Learning: A Review. *Electronics* **2021**, *10*, 999. https://doi.org/10.3390/electronics10090999

Academic Editor: Mohamed Benbouzid and Juan M. Corchado

Received: 5 March 2021
Accepted: 17 April 2021
Published: 22 April 2021

1. Introduction

In recent years, the huge advancement in information technology and artificial intelligence largely impacted intelligent autonomous systems, which are becoming ubiquitous. Such evolution includes Unmanned Aerial Vehicles (UAVs) consisting of fully autonomous, semi-autonomous, and pilot-based or remote control-based flying vehicles. UAVs are a class of aircraft that can fly without a necessary need for an onboard human pilot [1]. They are commonly known as drones, and they can fly with various degrees of autonomy: either under remote control by a human operator or autonomously by onboard computers [2]. These drones can fly within Visual Line of Sight (VLOS) for limited distances or Beyond Visual Line of Sight (BVLOS), covering far greater distances.

UAVs have been around for decades and were mainly used for military purposes. After that, they found their role in each field, either in industry, military, or even enter-

tainment. UAVs' initial spread started by their use in aerial photography applications, offering a new level of creativity in either pictures or videos. They help in monitoring some commercial and safety needs [3,4]. Expanding into new scopes, UAVs are used in conjunction with terrestrial sensors to gather data in places with no infrastructure networks, as they provide direct connectivity to the internet; UAVs play a role in monitoring hostage when dangerous locations such as in mining and energy production systems are visited [3]. Moreover, UAVs can perform initial assessment analyses for disaster areas and work as first-aid devices. Once a survivor is detected, the crew responsible for the mission tries to contact him to rescue him as fast as possible; in such cases, UAVs help detect surviving individuals and initialize communication between both sides [5]. Despite all of these successful applications for UAVs in real life, their benefits on the commercial level and its autonomous mode of operation were not sufficient to allow free-provided legality; some permissions are necessary to operate in such ways. Some regulations prohibit UAV operation beyond the line of visual sight or over specific announced altitude limits.

The emergence of UAVs is associated with the usage of radio signals that allow remote control of the vehicle. These vehicles were slow and incapable of flying for long distances, such as the Kettering Bug first flight that was done in 1918 with a speed of 80 k/h and for 120 km. Advances in the UAV technologies and mass production were made during world war II, which signifies a prominent step in developing unmanned aerial vehicles. After that, UAVs began to grasp the public's attention, and the industry started to develop for non-military purposes, and with greater capabilities [6].

Nowadays, unmanned aerial vehicles are different from the early models of UAVs, in being smaller and less expensive [7]. Moreover, they are designed to follow some standards that ensure the safety of their operations. For example, studies are now concerned with fail-safe operation, analysis of the exposure, analysis of hazard threshold, risk analysis, assessment, control, and management [8–11]. Exposures are the individuals or components affected by the event taken by the vehicle, while the hazard threshold is the limit after which hazardous effects begin to appear throughout the whole operating [12].

UAVs have been categorized according to many important aspects such as top-level configuration, limiting altitude, mean Take-off weight, autonomy level, and even ownership. First to be discussed is the classification of UAVs according to top-level configuration into fixed-wing, rotary-wing, and hybrid wing. Fixed-wing UAV type has a rigid wing with an airfoil that works by lifting the forward airspeed to fly, like regular human planes. This configuration supports longer endurance flights and loitering, provides high-speed motion, and maintains high payloads relative to the second configuration with rotary wings. Some of the issues associated with this configuration are its need for a runway to take-off/land as it relies on the forward airspeed, as mentioned before, and hovering tasks are not performed, as it must keep continuous flying until landing at the end of any trip. Second is the rotary-wing configuration type that presents maneuverability advantages using the rotary plates. Its rotary blades can produce aerodynamic thrust forces enough to fly without the need to airspeed relative velocities. Such characteristics allow this type of UAVs to handle vertical take-off/landing, fly in low altitudes such as in complex urban environments, perform hovering tasks. However, it cannot preserve the same payload maintained by fixed-wing configuration. Technically, this configuration is further divided into subcategories according to the number of rotors included; it includes single-rotor (i.e., helicopters) and multi-rotor (i.e., tri-rotor drones, quadcopters, etc.). Single-rotors present mechanical complexity and high cost as they can take-off or land vertically, maintain a relatively high payload, while the multi-rotor is much faster and able to hover or move around a target in a highly smooth way. The third is the hybrid configuration that has been raised as a particular type of aerial platform that sums the advantages of both known configurations, classified into Convertiplanes and Tail-sitters.

The second classification for UAVs takes into account the altitude they can dive in. The first level is Very Low Altitude, and it operates in altitudes less than 400–500 ft with the pilot always in visual contact with the vehicle. The second level is the Very Low Altitude

that operates in the same range as the first level with allowance for the aircraft to disappear behind the pilot vision. The third level is Medium Altitude that operates from Class A to Class E airspace, and the last is Very High Altitude that operates above FL600 and in Class E airspace. Another classification is made according to Mean Take-Off Weight (MTOW) and Ground Impact Risk [13,14]. It is satisfying enough as it correlates the kinetic energy imparted at impact, which is consequently considered the primary factor for safety, as discussed by Weibel and Hansman (2004) and Dalamagkidis et al. (2012) in [14,15].

Although previous classifications were helpful, others were conducted to group UAVs based on their autonomy. In 2005, Autonomous Control Levels (ACL) were introduced to measure the level of autonomy or the level of independence from human piloting involvement. Each level relies on three aspects: level of independence, mission complexity, and environment configurational level [16]. The first one is remotely piloted. This level is fully controlled by a certified expert getting visual feedback or sensory data. The second is remotely operated, named semi-autonomous. This level allows the vehicle to drive itself based on the decisions coming from observing the pilot. Last is a fully-autonomous level. At this level, the vehicle is provided with general tasks to perform, and it becomes capable of doing self-maintenance once a fault occurs. At this level, the owner only informs the vehicle about what to do without specifying how to do that task. Although the fully-autonomous level seems interesting, it may cause issues when performing prohibited or prohibited behavior to solve the task [17,18]. Further classifications regard the Ownership; either the UAV is owned by a public entity, like federal agencies, or by law enforcement, like industry and private individuals [19]. The international American Helicopter Society presents classifications, configurations, and propulsion concepts of UAVs (AHS) (American Helicopter Society (AHS) International: The Vertical Flight Technical Society https://vtol.org/store/index.cfm?killnav=1).

This paper covers the main three challenges facing UAVs: (i.) path planning, (ii.) navigation, and (iii.) control. Each of these elements includes many sub-challenges that need a high level of control to function as desired. Reinforcement learning algorithms are used to help navigation through unknown environments that have no mathematical model that is suitable to describe them. The article is organized as follows: The upcoming section discusses several reinforcement learning algorithms related to UAV and its learning behavior. Then, Sections 3–5 discuss path planning of drones, navigation, and control, with the help of the reinforcement learning approaches mentioned. Future work and recommendations of research directions are then presented in Section 7. The last section is a summarized conclusion of the main approach and findings of this research.

2. Overview of Deep Reinforcement Learning

Reinforcement learning (RL) is learning what to do—how to map situations to actions—to maximize a numerical reward signal. RL features an interactive intelligent agent with an explicit goal to achieve. Policy, reward, value, and environment model are the essential elements of RL. The policy defines the agent's plan of action (i.e., how the agent reacts to different environmental situations and how it translates the states to actions). Rewards are the numerical values given by the environment to the agent in response to a state-action pair. These reward values describe the immediate, intrinsic desirability of environmental states. The value function is the long-term version of a reward function, calculating discounted return starting from a specific state following a particular policy. The environment model represents the environment behavior that helps boost the algorithm performance by understanding the surrounding environment. For example, while DeepMind AlphaGo imitated human strategies and applied Monte Carlo tree search using three convolutional policy networks in a supervised manner, AlphaGo Zero did not imitate human behavior and did not see data from human games; instead, it played many games against itself to learn and thus developed strategies to win the game that is not known to humans, thanks to reinforcement learning.

An Agent, in the context of reinforcement learning, is the entity that is asked to take any action in the environment, given the current state and the past experiences. The main objective of any reinforcement learning algorithm is to let the agent learn quickly the optimal policy, which is represented by the symbol π, that achieves the targeted task accurately, and thus results in the highest reward value [20]. The interaction between the agent and the environment is shown in Figure 1, in which the agent (the drone plus the learning algorithm) decides to take action $a_t \in A$ from the pool of available actions (for example, move forward) at time t, and executes it in the environment (e.g., the drone moves forward). The result of that action is a change in the state $s_t \in S$ that makes the agent closer or not to its target (e.g., it flies to a given location). The reward $r_t \in R$ quantifies the good reward if the agent is closer or the bad reward if it is further away.

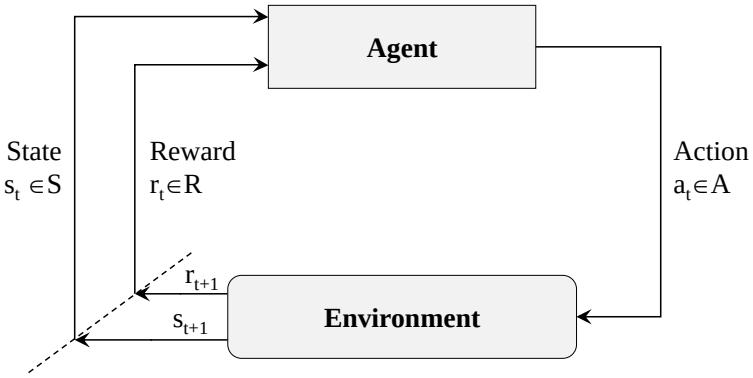

Figure 1. The interaction between agent and environment in RL.

Deep reinforcement learning (DRL) is a subfield of machine learning, which uses deep learning (DL) concepts with RL to provide an optimal solution based on experience. This experience is based on the iterations and evaluating a reward function in determining the best behavior for an agent.

Approaches in deep reinforcement learning can fall into three main categories, namely the value-based approach, policy-based approach, and model-based approach. In value-based reinforcement learning, the agent has the objective of finding the policy that maximizes a value function in the long run over a sequence of actions. Then, in policy-based reinforcement learning, the agent has to find the policy, leading to the optimum value for the objective function. This category is further divided into deterministic and stochastic approaches. The former policy applies the same action in any state, while the latter includes variations in actions based on probabilistic evaluations. Finally, model-based reinforcement learning depends on providing a model for the environment to the agent or asking the agent to learn a model of the environment to perform the tasks in that specific environment. It is not easy to directly compare the model-based and model-free reinforcement learners. Poole and Mackworth claim in their book [21] that model-based learners are typically much more efficient in terms of experience; many fewer experiences are needed to learn well model-free methods often useless computation time. In learning the environment model on its own, the agent will face some inaccuracies and imprecision that can further affect its policy and the required tasks. Thus, many approaches were proposed to integrate the model-free approaches with the model-based ones [22].

Reinforcement learning algorithms are classified from different perspectives, including model-based and model-free methods, value-based and policy-based methods (or combination of the two), Monte Carlo (MC) methods and temporal-difference methods (TD), on-policy and off-policy methods, which are presented in Figure 2 [23].

Despite the differences implemented in the three approaches, they share some important characteristics inherited from the concepts of deep reinforcement learning. The control implemented by reinforcement learning acts like a closed-loop one, while the reward represents the system's feedback. This reward is delayed, while many algorithms try to

decrease this delay. Moreover, the algorithms implemented in DRL feature a sequential decision-making behavior, with long-term rewards depending on the actions' sequence. A concept called credit assignment problem describes the DRL implementation's dependence on time as some actions show their consequences after some time and many intermediate states undertaken by the system [20].

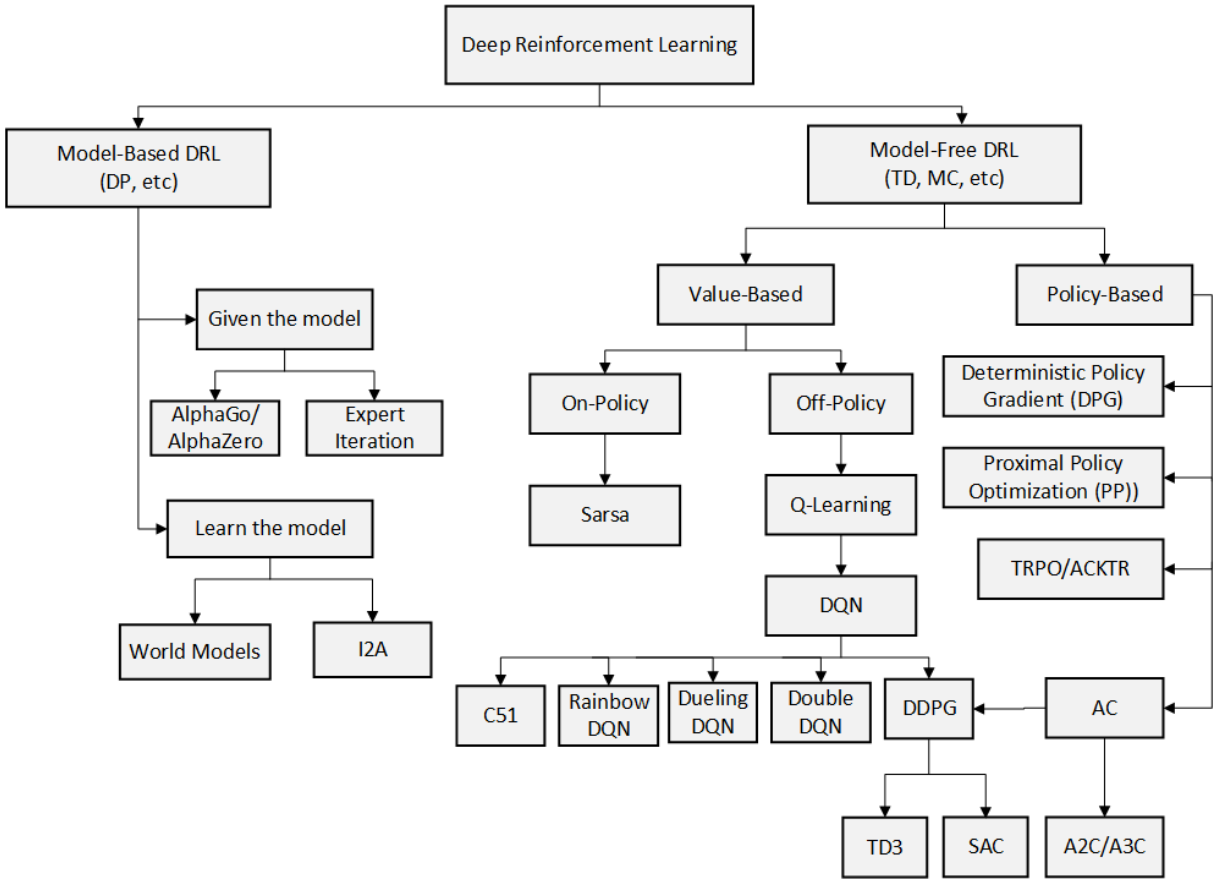

Figure 2. Taxonomy of Reinforcement Learning Algorithms. DP (Dynamic Programming), TD (Temporal Difference), MC (Monte Carlos), I2A (Imagination-Augmented Agent), DQN (Deep Q-Network), TRPO (Trust Region Policy Optimization), ACKTR (Actor Critic using Kronecker-Factored Trust Region), AC (Actor-Critic), A2C (Advantage Actor Critic), A3C (Asynchronous Advantage Actor Critic), DDPG (Deep Deterministic Policy Gradient), TD3 (Twin Delayed DDPG), SAC (Soft Actor-Critic).

Another classification for deep reinforcement learning is the categorization of techniques as positive and negative. The positive ones use favorable stimulation of the system to attract the agent for optimizing its behavior. On the other hand, the negative ones use undesirable stimulation to repel the agent from specific actions. As the whole system of a UAV is based on the interaction between the agent and the environment, achieving the optimal strategy requires high dimensional inputs and an accurate representation of the environment. Applying the RL algorithm to a UAV system can be considered a customized representation of Markov decision processes (MDPs) as the process requires four main elements: a policy, a reward signal, a utility function, and an environment [24]. Relying only on the conventional RL algorithms requires high quality and multi-dimensional inputs to generate an accurate description of the past state on the UAV. Those requirements form a gap due to the capabilities of the conventional RL. Integration between the deep neural network and the RL algorithms is developed to fill this gap.

The application of deep reinforcement learning (DRL) in unmanned aerial vehicle control was introduced to face specific challenges that were encountered in that field. DRL helps in the task of UAV control to work with model-free algorithms when the

UAV model is complicated to find, to account for nonlinearities in the system, to learn online actively how to achieve the target without being explicitly trained to do so and to work in environments unknown to the UAV. Figure 3 shows the taxonomy of different tasks encountered during UAV control and shows the suggested algorithm(s) for each problem. DRL opens the door for new challenging tasks in the UAV control area, such as controlling hybrid UAV safely and controlling a swarm of UAVs to achieve a specific task with minimum resources (i.e., minimum moves, time, energy, etc.).

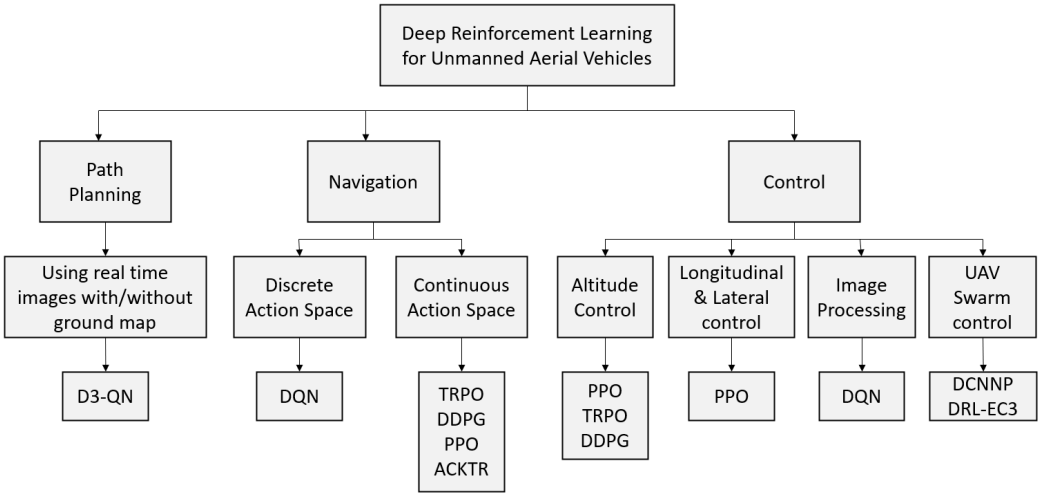

Figure 3. Taxonomy of Deep Reinforcement Learning Algorithms for Unmanned Aerial Vehicle Tasks.

There are some well-known algorithms in the field of deep reinforcement learning. Often, they are used as they are, or some researchers use them as the base algorithm on which some modifications are done to optimize the behavior. This optimization can decrease the computational time, make the system more energetically efficient, improve performance, etc. To start, the Deep Q-Network algorithm (DQN) has shown outstanding performance in solving complex problems. Using DQN in UAV control systems, such as path planning, navigation, and attitude control, is based mainly on knowing the current state space of the system (S), the reward value (R), the transition probability (P), and the utility function (V) [25]. Equation (1) expresses the Q-function in terms of those parameters.

$$Q^{\pi}(s,a) = R(s,a) + \tau \sum_{s' \in S} P_{ss'} V^{\pi}(s'), \tag{1}$$

where π is the policy value, s is the state, a is the action taken, and τ is the discount factor. The simulation is divided into time steps t, and at each step, a new reward value is calculated until reaching the final step T.

After obtaining the current Q-function, a sequence of steps is taken, as shown in Algorithm 1, to reach the optimal Q-function and policy value. Afterward, the maximum Q-function and the cumulative discount function are calculated from Equations (2) and (3) respectively. As illustrated in Algorithm 1, the first four steps are the initialization of the parameters in the DQN. To ensure a stable learning process, the target of the DQN is introduced in a specific structure; then, the action is derived based on the current state of the UAV [26].

$$Q^{\pi^*}(s,a) = R(s,a) + \tau \sum_{s' \in S} P_{ss'} V^{\pi^*}(s') = \mathbb{E}[r + \tau max(Q^{\pi^*}(s',a') \mid s,a)] \tag{2}$$

$$V^{\pi^*}(s) = max_{a \in A}[Q^{\pi^*}(s,a)] \tag{3}$$

Algorithm 1: Pseudocode for DQN-Based Training Method.

```
Initialize the Q(s,a,w) table with random weights w;
Initialize the DQN parameters with w⁻ = w;
Construct the DQN structure;
Start the environment simulator;
for episode = 1,2,...,num do
    Initialize all the beginning states s as zero;
    for t = 1,2,...,T do
        Obtain the received signal of the UAV;
        Obtain the instant reward value;
        Choose the action based on the given probability;
        Observe both the current reward value and the next state;
        Update the network parameters w of the DQN;
        Update the Q(s, a,w) table;
    end
end
```

Introducing the DQN in RL opened the gate to more improvements and innovative algorithms that seek the most stable autonomous navigation. As the DQN is based on updating and accumulating each state of the system for taking any action, that can cause correlations during the learning process. This is due to the accumulation in the DQN parameters and the fact that it uses randomly generated patches of experience [27]. In order to avoid this issue, Double DQN was proposed, as it uses different values each time to induce the action, as shown in Equation (4).

$$Y_t^{Double} = r + \gamma Q(s_{t+1}, argmax Q(s_{t+1}, a_{t+1}, ; \theta;)\theta^-). \tag{4}$$

Continuing the enhancing process on the conventional DQN, Dueling DQN was proposed, which relies on dividing the networks into value networks, and advantage networks [28]. The value network aims at evaluating the quality of the current and each state. However, the advantage network is related to the quality of the actions, as shown in Equation (5).

$$Q(s_t, a_t; \theta, \alpha; \beta) = V(s_t; \theta, \beta) + A(s_t, a_t; \theta, \alpha) - \frac{1}{A} \sum_{a^{t+1}} A(s_t, a_t; \theta, \alpha). \tag{5}$$

To take advantage of the two proposed techniques, a combination of them called Double Dueling DQN (D3QN) was created, based on both Equations (4) and (5) that are used to evaluate the target value of the D3QN as shown in Equation (6).

$$Y_t^{DDQ} = r_{t+1} + \gamma Q(s_{t+1}, argmax Q(s_{t+1}, a_t; \theta, \alpha, \beta); \theta^-, \alpha^-, \beta^-). \tag{6}$$

Another well-known algorithm in the field of DRL is the Proximal Policy Optimization (PPO). In a nutshell, this algorithm is based on replacing hard constraints with flexible ones, which are considered penalties. The new constraints, which are easier to handle, are used for finding a solution to a first-order differential equation that approximates the second-order optimization differential equation. The Kullback–Leibler (KL) divergence is used to determine the policy changes during each time step. Algorithm 2 shows the pseudocode for the PPO methodology [29].

$$Q^{\pi}(s, a) = \sum_t E_{\pi_\theta}[R(s_t, a_t)|s, a] \tag{7}$$

$$V^{\pi}(s) = \sum_t E_{\pi_\theta}[R(s_t, a_t)|s] \tag{8}$$

$$\hat{A}_t = Q^\pi(s,a) - V^\pi(s) \tag{9}$$

where $R(s_t, a_t)$ is the reward, $V^\pi(s)$ is the expected long term reward when the π_θ policy is followed while being in state s and $Q^\pi(s,a)$ is the expected long-term reward when the π_θ policy is followed while doing action a in state s.

Algorithm 2: Pseudocode for Proximal Policy Optimization (PPO).

```
Initialize the total number of iterations (Z), number of actors (N), and the time
  steps (T);
for iteration = 1 to Z do
    for actor = 1 to N do
        Run the policy π_θ on the old theta value, θ_old, for T time steps;
        Compute the advantage estimates Â_t for the time steps, as shown in
          Equation (9);
    end
    Optimize the objective function L(θ) with respect to θ and call it θ_opt;
    Define θ_old = θ_opt ;
end
```

Some variations to the PPO algorithm are proposed and implemented in various problems. For example, the penalty can be adaptive for fast performance, or the objective can include two policies: the current policy that the algorithm tries to optimize and the old policy used to collect samples to test the current policy.

Moreover, Deep Deterministic Policy Gradient (DDPG) comes into play as a crucial and widely-implemented deep reinforcement learning algorithm. In this algorithm, the network outputs are used directly as actions implemented by the agent instead of probabilities evaluating the actions. Moreover, a target network is used in this algorithm to ensure convergence by using the delay in time for model evaluation. Like many other algorithms, the DDPG uses a replay buffer to update the parameters of the networks. The block diagram of DDPG is shown in Figure 4 [30], and the pseudocode for the DDPG methodology is shown in Algorithm 3 [31].

$$a_t = \mu(s_t|\theta^\mu) + N_t \tag{10}$$

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'}) \tag{11}$$

$$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q)^2) \tag{12}$$

$$\nabla_{\theta^\mu}\mu|_{s_i} \approx \frac{1}{N}\sum_i \nabla_a Q(s,a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|s_i \tag{13}$$

$$\theta^{Q'} = \tau\theta^Q + (1-\tau)\theta^{Q'} \tag{14}$$

$$\theta^{\mu'} = \tau\theta^\mu + (1-\tau)\theta^{\mu'} \tag{15}$$

The algorithms used are numerous, and they are suited to different problems based on their performance and the solution requirements. In this context, the Asynchronous Advantage Actor-Critic (A3C) is a great algorithm to be used in deep reinforcement learning. This algorithm is model-free and on-policy, and it features continuous action and state spaces. This technique includes one global actor-critic network along with many local actor-critic networks. The local networks use the information from the global one to start learning, and in return, they update it afterward by their gradients. The loop of Actor–Critic is shown in Figure 5 and Algorithm 4 shows the pseudocode for a single Advantage Actor–Critic Agent methodology [32].

$$R = r_t + \gamma R \tag{16}$$

$$d\theta' = \nabla_\theta log(\pi_\theta(a,s))(R - V_w(s)) \tag{17}$$

$$dw' = 2(R - V_w(s))\nabla_w(R - V_w(s)) \tag{18}$$

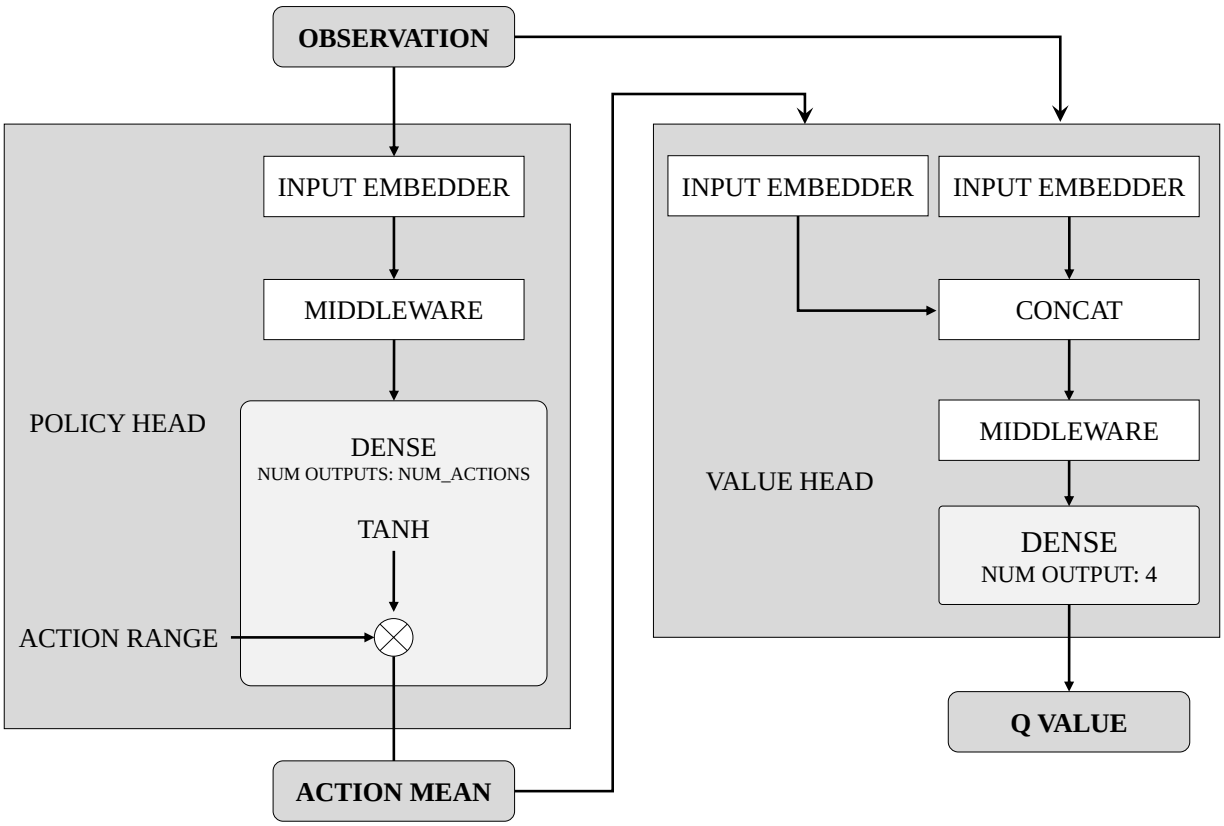

Figure 4. Deep Deterministic Policy Gradient (DDPG) block diagram.

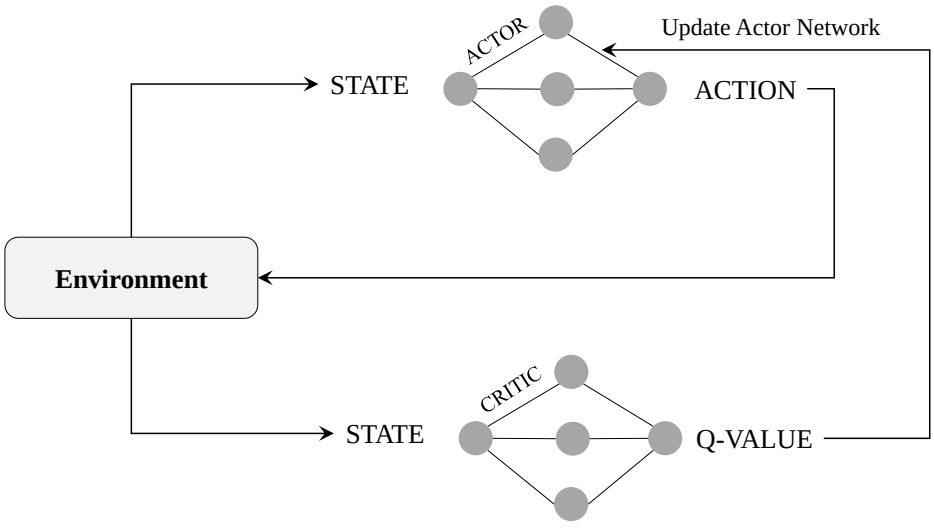

Figure 5. The loop of the Actor–Critic Algorithm.

Algorithm 3: Pseudocode for Deep Deterministic Policy Gradient (DDPG).

Initialize the critic network $Q(s)$ and the actor network $\mu(s)$ randomly with weights θ^Q and θ^μ, respectively;
Initialize the target networks Q' and μ' with weights $\theta^{Q'}$ and $\theta^{\mu'}$, respectively;
Initialize the replay buffer R;
for *episode = 1 to M* **do**
    Use a random procedure N to explore actions and determine the system state s_1;
    for *t = 1 to T* **do**
        Use Equation (10) to get action;
        Implement the action and find the corresponding reward r_t and state condition s_{t+1};
        Save the initial state condition, new state condition, reward and action in R;
        Sample a random N transitions from R;
        Use Equation (11) to find the variable y_i;
        Minimize the loss function L using Equation (12) and use it to update the critic network;
        Find the gradient using Equation (13) and update the actor network;
        Update the target networks weights $\theta^{Q'}$ and $\theta^{\mu'}$ by using Equations (14) and (15);
    end
end

When the deep reinforcement learning methods are applied in practical, real-life problems, some challenges appear that affect the accuracy and performance. As mentioned in [22], some limitations prevent us from applying the agent to the natural environment, so we apply it to a simulating environment due to cost or safety concerns. These circumstances lead to the dilemma of possibly having inaccuracies in the simulation and/or being unable to obtain new observations to boost the learning process. Although being tricky, these problems can be controlled through thoughtful design of the simulating environment and the model to allow its generalization. Another challenge discussed in [20,22] is the balance between exploitation and exploration to maximize the algorithm's efficiency. One approach is to use a significant probability to describe the ratio of exploration to survey the environment, then decrease its value with time to focus more on exploitation. In addition to the challenges that come from using reinforcement learning, some other challenges are related to the flying and controlling of UAVs themselves, such as the trade-off between payload and travel time, the weather conditions, and the limitations of the sensors used [33]. However, even these problems can be controlled through deep reinforcement learning that corrects the behavior in the presence of disturbances and optimizes the solution to get the maximum reward from the trade-off.

To better understand the different algorithms in deep reinforcement learning and evaluate their performances, the actual implementation is needed. In this paper, the implementation of DRL in UAV systems is analyzed. Because UAV control areas are vast and include different DRL problems, they are divided into three sections. The first one is about UAV Path Planning, the second is about UAV Navigation, and the third is about UAV control in terms of lateral and longitudinal movement and between-UAVs communications.

Algorithm 4: Pseudocode for a single Advantage Actor Critic Agent.

```
Initialize randomly the global parameters w and θ;
Initialize the counter, T, with value 1;
while T < T_max do
    Synchronize the local parameters, w' and θ' with the global ones w and θ;
    while t < t_max or s' is not terminal do
        Do the action, a = π_θ(a, s), and find the corresponding state s' and reward r_t;
        if s' is terminal then
            The variable R is equal to r_t;
        else
            The variable R is updated according to Equation (16);
        end
        Use Equations (17) and (18) to update the policy gradient and value gradient, respectively;
        Make the state s equal to the new state s' and increment the variable t with 1;
    end
    Update global w and θ using dw' and dθ';
    Increment the t variable with 1;
end
```

3. Reinforcement Learning for UAV Path Planning

Path planning is a very important research topic to be considered as it correlates with the level of autonomy of the vehicle. It is correlated with autonomy level, but it also impacts built-on components, guidance, functionality, and endurance. Path planning is an effective umbrella term that includes time control for a trip, object avoidance, and many other vehicle challenges. Several researches have been conducted, presenting various techniques that use reinforcement learning doing the path planning task or covering part of its sub-challenges. UAVs are most commonly placed for missions in unknown or partially observable environments where no exact mathematical model can be defined. Therefore, reinforcement learning and deep learning are combined and used to allow vehicles to learn their paths autonomously. The application of advanced learning algorithms allows the UAV to go through changing environments without the risk of collisions [34].

In fact, reinforcement learning algorithms have been widely used in different areas of research related to UAV applications. For instance, [35] introduced an RL algorithm that enables the UAV to have direct and continuous interaction with the surrounding environment. The paper proposed a combination of DRL and an LSTM network to converge the speed of the used learning algorithm. Another one designed in [36] is used in object avoidance and target tracking via the usage of a reward function and penalty action to achieve a smoother trajectory. Additional research was led by Koch et al. uses the RL algorithm to obtain accurate attitude control [37].

Seeking smoother flight path, RL has been used with a variety of optimizers such as Grey Wolf Optimizer (GWO) in which the algorithm mimics the behavior of a pack of wolves while hunting [38]. The GWO is an advancing metaheuristic algorithm that simulates the hierarchy of leadership in the grey wolf pack and the process of surrounding and attacking their prey. The algorithm has a bionic structure that can be flexibly implemented. In [39] a novel UAV path planning algorithm is introduced to generate a refined and optimal path for the UAV. The proposed technique combined the advantages of RL and GWO to promote the four processes: exploration, geometric adjustment, exploitation, and optimal vehicle adjustment.

With a large amount of captured data that needs processing, there are gold-standard approaches for avoiding obstacles that use computational techniques such as Simultaneous Localization and Mapping (SLAM) and Structure from Motion (SFM) [40]. SLAM algorithm

utilizes data from one or more sensors to build a map for the surrounding environment. At the same time, it uses this data to update the UAV position and path plan. Another approach for obstacle avoidance is introduced that uses a depth camera to localize the attributes of an object. The data in the research is trained through a novel developed algorithm that uses DRL and imitation learning [41].

In urban environments, planning the optimum path for a UAV requires a large set of data collected from distributed nodes such as sensors and other changing parameters like the number and locations of these nodes. In [42], a learning algorithm is developed that creates a network architecture base on Internet of Things (IoT) devices. A DDQN network is trained while taking into consideration parameters such as allowable flying time for the autonomous drone, sensor positions and numbers. The proposed network enables the UAV to adjust its behavior and to make movement decisions in changing scenarios.

Regarding SFM, it uses optical flow sensors to generate a depth map and 3D structure for the surrounding environment. Both methods require a prefixed path for the UAV to follow; accordingly, UAV is required to hover while computing the environment and planning the path and follow this path thereafter [43]. It means that those methods cannot be used for real-time applications of obstacle avoidance. Although some enhancements have been done to use the SLAM technique on the fly [44], they neither include non-stationary obstacles with unpredicted motion nor detecting un-textured walls that usually exist in indoor environments [40,44].

For obstacle avoidance, Ref. [45] proposed a self-trained UAV that can avoid moving and static objects in a 3D urban environment. The research used a learning algorithm based on Deep Deterministic Policy Gradient (DDPG) to help the vehicle reach the target with zero collisions. The algorithm uses a reward function that minimizes the distance between the UAV and its final distention, along with a penalty for each collision with an object.

Another technique designs the obstacle avoidance challenge as a model-free reinforcement learning problem as in this case, the environment is not fully known in advance [44]. A deep reinforcement learning approach is used for indoor environments by training UAVs for navigation based on 3D hallway environments. Many images for such environments are collected with various lighting levels, furniture placements, and wall textures. A massive data set is generated, and the deep Q-network learns UAV motion on these photos. Although many images are included in the training dataset, it is not enough to satisfy basic levels of accuracy. Moreover, this method is not preferred as it does not mimic human learning behavior for obstacle avoidance, which means that there is a better method to follow.

Despite the partial failure of such a method, it inspired researchers for the usual way of the success of mimicking nature and human behaviors for solving problems. So, researches focused on human behavior in avoiding obstacles until it is clarified as follows. Human does not have full access to the environment, but it can solve challenging problems. It relies on human memory that stores all relative information and recalls it to decide the following suitable action for each scenario. UAVs have a similar problem of partial observability that requires a notion of memory to enhance position planning at each instant [44]. For instance, a UAV moving toward a corner will gather the information that there is much space in front of it, although it is not the case for all sides. Without the past gathered information about sides and previous navigation, the UAV will proceed until it crashes with the walls. So it becomes an aim to design an algorithm capable of combining information collected over some time in order to enhance real-time navigation decisions [44].

A method is proposed to safely navigate an indoor environment dealing with stationary and dynamic obstacles in unstructured and unknown environments. It uses a monocular camera to provide an RGB image instantly as an input and uses the conditional generative adversarial network (cGAN) to predict the desired depth information required for UAV to take action. The UAV is required to take any action at every instant t, which will affect future states and even future decisions for UAV navigation. In other words, the

realization of the upcoming state is probabilistic, confirming that navigation by obstacle avoidance is a sequential decision-making problem.

The model can be described by seven variables (S, A, P, R, Ω, O, δ), where S is a set of environment states or state-space; it represents a collective descriptive property of the environment. A is the set of possible actions, action space. P is a probability function modeling the evolution of each state based on current and previous actions. R is the reward function that serves feedback to UAV for the action chosen; for example, if an action leads to correct navigation avoiding a faced object, it will receive a positive value for reward function while it will be negative if the action is taken results in collision between the UAV and the object. Ω is a set of real observations, which is the percentage of the true state, s. O is the conditional probability distribution on observations and δ is the discount factor [44]. This model translates the problem to maximize the expected sum of discounted rewards. It uses a deep recurrent Q-network with temporal attention, utilized by reinforcement learning for effective object avoidance in unseen environments. Q-learning uses an action-value function to know how to maximize a reward value for a specific action. In other words, the agent will work under the policy formed, and the optimal one will be configured by choosing actions with the highest reward values [5].

A similar model of reinforcement learning has been used in [34] on a position controller in a trial of enhancing UAV behavior [34]. It relies on the vehicle's current position and learning model to decide the following action to perform for the upcoming state. The generated position will be input to position controller that gives the action for propellers controller to, then, form thrust force enough to drive the vehicle to required new position [34].

Another research has tried to combine the sub-challenge of object avoidance with another task, arriving at the end target through the shortest path and minimum time interval—learning of the flight path conducted using Q-learning algorithm such as in previously discussed research. This main challenge of this research is the computation capability required to plan the path while avoiding any random obstacle; a calculation speed guarantee is a must. Accordingly, a separate unit is required to perform all computations and calculations, and the final learned results are reflected in UAV after that [5]. Simulation for a simple indoor environment is constructed under this research containing four main elements: walls, obstacle, start position, and target position, each with reasonable reward value. Then, four different maps are generated with different sizing, obstacle numbers, and elements, placing [5].

A novel method has been proposed in [46] that discusses different sub-challenges that UAV has to face. It discusses an interference-aware path planning scheme that allows minimization interference with a ground map and wireless transmission latency caused while sharing online mission-related data [46].

The problem is represented as a dynamic noncooperative game given the UAV plays the user's main role. A deep reinforcement learning algorithm relying on echo state network (ESN) cells is used as it guarantees to reach a subgame perfect Nash equilibrium (SPNE) when converges. The main contribution is this reinforcement learning framework used for optimizing the trajectories of the online-connected UAVs. Each UAV is required to learn its path with respect to the ground network and other moving UAVs. This technique allows UAV, the user, to decide the next step of the plan, including location and transmission power level. The main features of this method are adaptation and generalization. UAVs do not need to watch the environment to decide the next action. Instead, it can take the decision based on a reward from the previous state.

Recently, the use of deep reinforcement learning in UAV path planning challenges becomes familiar to some extent. This opened a way for researchers to perform more complex situations moving toward real-time path-planning, especially in dynamic environments. In 2019, Chao et al. proposed a deep reinforcement learning approach for real-time path planning based on information gathered about the global situation. This is more challenging due to the consideration of potential enemy threats with the main planning

task [47]. This has a very important political application in anti-terrorist tasks. In the scope of this research, the threat is enemy radar detection of the vehicle when it is within a specific range, and surface-to-air missiles can affect a vehicle's probability of survival. Therefore, UAV must solve its task while ensuring self-safety at the same time [47].

The model environment is developed, including enemy entities using a software tool named STAGE Scenario. This tool has a rich database, including battlefield entities such as radars and missiles [20]. Necessary attributes such as UAV, target, and enemy locations are selected [47]. The situation assessment model is then constructed with the main collected data of vehicles' position and enemy unit position; the enemy unit consists of the radar and surface-to-air missile. Once the UAV is detected inside the radar range, calculations relating the distance between enemy and vehicle and a maximum radius of the killing zone are done, and the missile is projected to destroy the UAV. A Dueling Double Deep Q-Networks (D3QN) algorithm is used to train the vehicle for the main task of path planning.

At the first of each training episode, initial network parameters are set, and the UAV is set to its initial position in the STAGE Scenario. Second, the situation map is generated according to the proposed situation assessment model as the currently active state. An action is selected according to the policy and executed. Once the action is executed, a new situation map is updated based on newly-obtained situation data, and immediate reward value is observed. According to memory, these new data are saved, replacing the oldest version saved [48].

4. Reinforcement Learning for UAV Navigation

Seeking the most stable control for a UAV, recent researchers have focused on using deep learning algorithms to achieve that. In [49], an experimental study is done on a UAV using different reinforcement learning and aims to divide those different algorithms into two main categories: discreet action space and continuous action space. The study started with an intensive comparison between the various methods used in drone navigation, such as supervised learning, unsupervised learning, reinforcement learning, and mentioning the advantages and the limitations for each of them. Reinforcement learning is based on the agent being trained to navigate and avoid obstacles through trials and errors. This characteristic is helpful as the agent will start learning by itself as soon as the training environment is ready.

The paper started using RL by deriving the equations for the sequential decision making in which the agent interacts with the surrounding visual environment by dividing it into discrete time steps. The equations illustrate how the time steps are interpreted to actions that the agent has to follow in order to send a reverse reward signal. The aim is to maximize this reward signal in RL through the discrete action space and the continuous action space. In discrete action space, the proposed agent decides to follow a policy in the form of greedy learning through choosing the optimum action based on a given state value. This value can be calculated in high-dimensional data, such as images, using a deep Q network (DQN). Given the issues that faced DQN, the paper proposed a new algorithm to solve those issues. The proposed algorithm combines the Double DQN and the Dueling DQN and calls it Double Dueling DQN (D3QN). The algorithm proved during experiments a robust ability in removing correlation and optimizing the quality of the states. Using the discrete action space, the study used a simulation platform called Airsim that generates its graphics from the Unreal Engine to help create a realistic simulation environment. Although the simulation provides perfect woodland, it does not provide complex paths for the UAV as all the trees within the simulation are planted on plain ground [50]. To solve this issue, the study created a new environment that includes different obstacles like solid objects in cubes and spheres, etc. In order to generate the optimum path for the drone, RGB and depth sensors were used along with CNN as inputs to the RL network. All the previously-mentioned DQN algorithms were used to generate the optimum behavior based on the following five moves: forward, left, right, down, and up.

The continuous action space learning also depends on obtaining the maximum reward value through different algorithms such as Trust Region Policy Optimization (TRPO), Deep Deterministic Policy Gradients (DDPG), Actor–Critic using Kronecker–Factored Trust Region (ACKTR), and Proximal Policy Optimization (PPO). In the proposed model, the continuous action space uses the U-Net-based segmentation combined with a policy gradient algorithm to solve manual data labeling as the RL training process replaces the labeling. The proposed combination used two learning processes: a segmentation map generated by the visual scene and Actor–Critic RL to control the UAV.

This study's experiment was designed to compare the training results and processes for the algorithms used in continuous and discrete action spaces. The comparison also included the results from the human piloting with various skills. The comparison between algorithms was performed in three different environments: woodland, block world, and an arena world. Regarding the discrete action space, the D3QN showed outstanding performance compared to the other used algorithms. As for the continuous action space, the ACKTR algorithm achieved smooth and stable continuous trajectories. The comparison between the algorithms and the human piloting was made by recruiting different pilots with various experience levels. The comparison relies on the RL performance and the human pilots in the three different virtual worlds. It showed that the expert pilot was the fastest in the woodland, while in the block world, the expert pilot and the ACKTR achieved almost similar results. ACKTR was the fastest to reach the arena world's goal, and the expert pilot came in second place.

Another application for RL was studied in [51] as the algorithms were used to achieve the optimum drone delivery. The study defined drone delivery as the path the drone takes to reach the destination while avoiding the different obstacles. Due to the outdoor applications for drones such as surveillance and delivery tasks, accurate navigation can not be achieved through a global navigation satellite system (GNSS) [52]. The outdoor tasks require the drone to avoid the changing obstacles, which can be achieved through the implementation of artificial intelligence techniques [34]. The study focuses on using different DRL algorithms that make the drone able to reach the target through an optimal decision-making system based on the reward value [53]. In this study, DRL algorithms are presented in theoretical manners and their application to the current delivery issue. The paper also presents new architectures to solve the problems in the current solution. The RL in this study represents the decision-maker as the agent and the surroundings as the environment. The interaction between the agent and the environment is performed in discrete steps, and each step represents a state in the state-space model.

The first proposed algorithm is the DQN, a combination between the Q-network and neural network algorithms [54]. The DQN mainly uses CNN to reach optimal action-value. The algorithm's idea was to use a periodic update to adjust the action-value towards the target-value, which reduces the correlation factor. Another proposed idea was to use experience replay, which a biologically inspired mechanism [55], and it is based on randomizing the data and removing the correlation in the different states. The Double DQN algorithm was also proposed within this study to produce positive bias through decoupling action. This study's experiment was done using a virtual environment, which is more like a realistic one to consider safety procedures. The state-space representation consisted of three different states and a neural network for each state. The three networks were the original convolutional neural network (CNN), the 2-dimensional version of Joint Neural Network (JNN-2D), and the modified JNN-3D to cover the drone's vertical movement. The three models' training took nearly 40 h for each; the authors aimed to improve training efficiency by using checkpoints. At the end of each training, two stored values are returned; the one for the network's weight and the other is the checkpoint representing the best reward value reached through the training. After completing the CNN training with and without applying the checkpoints, the results showed that the existence of checkpoints increased the number of successful action episodes while decreasing the number of unsuccessful ones. As for the JNN results, reaching the number of successful episodes took nearly half

the sample number that by CNN training. The reward value reached by JNN was higher compared to CNN with faster stabilization.

Most of the researches on UAVs focuses on implementing the training and optimization algorithms on a simulation platform. So, in [56], the authors focused on implementing RL algorithms on real hardware to make it follow an optimum path in an unknown environment. The paper started with addressing the main problem: making the UAV follow an optimum path by taking the desired action. Assuming that the current state of the UAV is S_k, the next state is S_{k+1}, and the thrust force is τ. The desired position is taken as the input to the position controller; then the controller will send signals to the motors to generate the thrust force that drives the UAV to the desired position. Figure 6 shows the block diagram for the controller addressed in this study. Most of the researches on UAVs focuses on implementing the training and optimization algorithms on a simulation platform. So, in [56], the authors focused on implementing RL algorithms on real hardware to make it follow an optimum path in an unknown environment. The paper started with addressing the main problem: making the UAV follow an optimum path by taking the desired action. Assuming that the current state of the UAV is S_k, the next state is $S_k + 1$, and the thrust force is τ. The desired position is taken as the input to the position controller; then the controller will send signals to the motors to generate the thrust force that drives the UAV to the desired position. Figure 6 shows the block diagram for the controller addressed in this study.

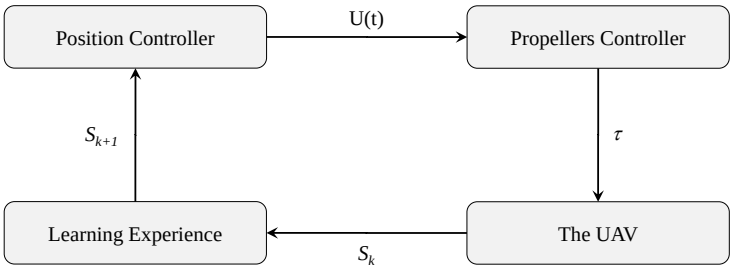

Figure 6. The proposed RL model.

The proposed design was to implement a standard PID controller to ensure stable navigation. The controller takes different learning parameters as inputs, along with the values of the three gains. The controller observes the current state of the UAV and updates the next state based on the accumulating reward equation. The controller's derivative component is mainly responsible for decreasing the settling time and the overshoot, and the integral component decreases the steady-state error, yet it can cause overshoot [57]. The tuning process took place to reach the optimum values for the three components, and the steadiest trajectory for the UAV was achieved when eliminating the integral component. To ensure the achievement of the desired output from the proposed algorithm, two steps were taken. First, a simulation was done on MATLAB by setting a bounded area for the drone. The UAV should travel from a certain point to another through the shortest path while avoiding obstacles. The training process took around 39 episodes to reach the optimum sequence of actions while increasing the reward and decreasing the penalty. The second step was implementing a quadrotor Parrot AR Drone 2.0 and the Motion Capture System software. Controlling the drone and implementing the proposed PID algorithm made the learning process computational expensive, and it took more time than the batteries can stand. To solve this issue and make the learning process easier and more reliable, the authors created a GUI on MATLAB that saves the data from each episode and shows the current states of the drone in real-time. Saving the UAV information along with the taken steps allows the learning process to continue where it stopped in case of any failure in the hardware. The experiment in real-time was performed using identical parameters to the ones used in the simulation, and the drone was not introduced into the environment before the experiment. The UAV followed the same behavior to simulation as it took 38 episodes

to reach the optimum sequence of actions. The learning process decreased the number of steps needed to reach the target from 100 steps to 8 steps in the last episode.

Another emerging application for UAVs is Search and Rescue (SAR) missions, where the UAV is required to navigate through an unknown environment and surrounded by hazardous risks and obstacles [58]. UAV usage in this field saved the human rescuers from entering dangerous environments and minimized the catastrophic effects caused by sending humans to perform missions in those environments [59]. Ref. [60] conducted experimental research to implement RL with a function approximation algorithm on a UAV to perform a SAR mission. The paper provided a detailed explanation for the problem and the implementation steps that include the mathematical model, the simulation, and the hardware implementation. The proposed experiment in this study considers a UAV performing a SAR operation, locating an immobile human in an unknown bounded environment. The study assumes a static location for the human, and a quadcopter UAV with an Ultra-Wide Band (UWB) is used for the mission.

By assuming that the UAV can localize itself with the environment and measure the distance to the surrounding obstacles using onboard radar, the RL algorithm can be trained using the system sensors' data. The algorithm of training the RL is the Q-learning, which is based on maximizing the reward signal. In this study, Q-learning helps the agent calculate the optimal value for the reward signal and then take the best-related action. The values related to each state of the system are recorded in a Q-table. The system's states are represented with the UAV position within the map; the authors used a fixed altitude for the UAV and divided the map into a discrete grid. To ensure the convergence of the Q-table, all the states and actions must be updated continuously. The continuous update of the Q-table values could lead the size of the table to grow in a way that will make it hard to implement the algorithm in reality due to hardware memory size and battery limitation [61].

To overcome this issue, the Q-table size has to be reduced yet still represents each state's values. This guarantee is achieved using one of the function approximation techniques called Fixed Sparse Representation (FSR). After implementing the algorithm on the data generating process, a PID controller was designed to ensure stable hovering and navigation towards the target. The design process started with the controller's three main components, and then the derivative gain was increased while eliminating the integral gain as it caused overshooting. The implementation was done using MATLAB for applying the original Q-learning and the approximated Q-learning. The simulation results showed that it took about 160 episodes using the original Q-learning to train the UAV to find the optimal route to the human target. Using the approximated Q-learning took only 75 episodes for the UAV to reach the human without any collisions. The realistic implementation was done using a quadrotor Parrot AR Drone 2.0, and a combined PID and approximated Q-learning was applied to the controller to manipulate the linear and angular speeds. The experiment took place in a closed room with no prior knowledge of the drone about the obstacles. The testing showed similar results to the simulation as the drone took the exact number of episodes to reach the optimum sequence of actions to reach the target.

Research by 2020 started considering dynamic obstacles that may face UAV while navigating, especially in realistic outdoor environments. All real obstacles, such as humans, cars, birds, and even other drones, are moving and have different speeds. Tai and Liu's studies first developed a model-free Deep Reinforcement Learning (DRL) algorithm based on a convolutional neural network [62]. This model allows a car to navigate an enclosed room with some static obstacles and some sparse dynamic obstacles. Although the consideration of dynamic obstacles succeeded, it had several limits as it only performs in simple indoor rooms, which is previously known, and obstacles have very limiting characteristics which are never guaranteed in a real environment.

Some main points were then highlighted to describe the research gap best not to deal with dynamic obstacles. First, the frequent collisions at the very start of training collect many penalties making the reward very small, which blocks neural network convergence and limits agent exploration capabilities. Second, a pure enhancement was required to

predict obstacle direction and speed to calculate the required drifting without forgetting about the main target; this needs real-time computation.

Considering the idea of decoupling and cutting the problem into sub-problems, the paper raised in 2020 discussing the consideration of high dynamic obstacles while reaching the target through an optimal path [34]. This research contributed in three main parts: first, a distributed DRL framework with two sub-networks, the Avoid Network and The Acquire Network, The avoid network job is to predict obstacle motion and steer away seeking collision-free trip while Acquire Network job is to reach the target through an optimal path. Both networks have shown more efficient results as each network has a denser rewarding at the very start by solving system confusion between the two jobs. The second contribution was integrating the results of both networks and choose a specific action to be finally implemented in real-time [34]. The third contribution was necessary to test the proposed approach; the authors constructed a simulation platform to prove the efficiency and robustness of the solution considering different obstacles with different speeds mimicking humans, birds, and cars. All obstacles are given random drifting values through the trip of UAV. UAV uses all of its sensors, cameras to explore the unknown environment, detect obstacles and probability of collision, deciding to move through the planned path or drift away according to the upcoming obstacle. This approach boosted the situation of UAV navigation in the previous years.

5. Reinforcement Learning for UAV Control

The area of controlling the UAV or a swarm of UAVs to perform a particular task faced in UAVs applications is very complex. This mission's complexity lies in its nature of facing changing operational and environmental conditions, its complex dynamics, its need to be energetically efficient, and its vulnerability to disturbances, sensor noises, and unmodelled dynamics. Nevertheless, there are many tasks and applications with noticeable variations in which a stable and accurate UAV control is needed. Classical control was first used; however, its performance was not satisfying. The main reason for such low performance lies in the dependence of this control scheme on linear dynamics. Thus, the controller can operate well on the UAV over a small set of conditions, taking into account its nonlinear dynamics. Then, machine learning with offline training was introduced. This also marks the onset of using reinforcement learning (RL). Although RL gives promising results when applied in a UAV controller, the need for higher accuracy and better robustness led to the design of deep reinforcement learning (DRL) control methods. Therefore, DRL opens the door for new control tasks that were unachievable with the previous RL algorithms.

Controlling the UAV, apart from navigation and path planning tasks, is still a broad concept that includes many different tasks and approaches. In this context, this section is dedicated to discussing significant contributions in UAV control from different points of view, ranging from attitude control of a single UAV to controlling the communication channels between multiple UAVs. In this context, deep reinforcement learning (DRL) is helpful because it allows online real-time learning and a model-free platform for control. It is essential to highlight that, to this aim, DRL must include several well-known policy-based model-free methods such as deep deterministic policy gradient (DDPG), trust region policy optimization (TRPO), proximal policy optimization (PPO), and so on. As expected, every algorithm has its benefits and drawbacks that make it suitable for some applications and not others.

It is worth mentioning that flying an autonomous UAV generally requires two control loops. The inner one is specialized in actuating the maneuvers by controlling the attitude, speed, and so on, while the outer one is for trajectory plans and communication needs. This outer loop can also be used in swarm managing. To start with single UAV controlling, the problem of attitude control is analyzed. In [29], DRL was used to control a fixed-wing UAV, especially to account for the nonlinearities of the dynamics and the coupling of lateral and longitudinal control, which cannot be ignored in such a UAV model. Proximal policy optimization (PPO) algorithm was used in the controller design to provide a stable flight

despite the disturbances and the different initial conditions and achieve the desired flying speed, roll, and pitch angles. The PPO algorithm was based on the Actor–Critic concept, and it enjoys a low computation time, which makes it a good candidate for a variety of tasks in the UAV controlling context. The actor-critic concept is based on using two separate networks. The actor-network estimates the optimal behavior, while the critic network estimates the reward of the behavior and uses rewards to train the actor. For the attitude control proposal made in [29], the reward function depends on the distance between the current state and the desired one. Another contribution of this research was to provide the neural networks with a normalized set of data to decrease the network's computation time to do the scaling itself. However, all the successful promising results of this work were done in simulations, which still raise concerns about the reliability of the proposal's real application, as mentioned in the paper.

Another attitude control problem was tackled in [37], where all Deep Deterministic Gradient Policy (DDGP), Trust Region Policy Optimization (TRPO), Proximal Policy Optimization (PPO), Proportional-Integral–Derivative controller (PID) are used, and their performances are compared. The interaction between the DRL agents and the environment was defined using Markov decision processes (MDP). Under this definition, a new state's probability is determined based on the current state and action. The results reveal that the PPO algorithm was the best-applied controller design than all other tested options, although the PPO algorithm and TRPO algorithm are known to have some similarities, as they are examples of policy-based gradient-based algorithms with better performance than natural gradient policy. To add to its advantages, PPO outperforms TRPO and has a more straightforward approach in implementing and tuning this task. Results were obtained from simulations and compared using definite metrics such as rise time, peak, error, stability, etc. The results were obtained from episodic discrete tasks; however, the research shows the method's availability to be used in continuous tasks.

Both [29,37] used PPO for the attitude control problem, even though the former applied it on fixed-wing UAV and the latter applied it on a quadcopter. This marks the suitability of the PPO algorithm for this control task, which falls under the umbrella of the UAV flying system's inner loop control. One great thing about research in this area of UAV control is that attitude control and all lateral and longitudinal controls are crucial to all UAV systems, with all the varying tasks they can perform and all the flying conditions under which they can achieve the tasks.

Moving to a more challenging control scheme, hybrid UAVs control has been studied by different researchers. Moreover, with the widespread use of deep reinforcement learning methods, such a learning approach was applied to hybrid UAVs, which are a particular type of UAVs combining the rotor concept with the fixed-wing concept to take advantages of their different characteristics, such as vertical landing and take-off (VTOL), long-endurance, energy efficiency, and so on. In [63], the hybrid flying system consists of 13 state variables; four of these state variables are convolutional integral error terms that are required to raise the accuracy of linear velocity and yaw angle. The use of deep reinforcement learning in this control problem to drive the vehicle was essential to enable real-time learning of the environment and collision avoidance practices that ensure the system's safety. In this context, the Proximal Policy Optimization (PPO) algorithm was again used for longitudinal and lateral control of that complex coupled system under the disturbances and noises presented in its operation. The paper's contribution was in using a generalized control scheme that can be used to multiple UAVs and can operate under a wide range of dynamics and conditions.

Moreover, the state variables are chosen carefully to include the mentioned four integral terms, which account for real application in an actual hybrid UAV. In other words, such a controller can be applied directly without additional terms or tuning to real UAV and succeed in its operation, similar to applying the controller to a stimulating environment. As known in DRL, a reward function is used to critique the available actions to determine the best next action. The choice of reward function depends on the controller

designer and should reflect the system objectives. So, in [63], the reward was based on the energy efficiency, stability of flight, integral error, velocity tracking error, and finally, orientation error.

Although it includes complicated advanced modeling and control, hybrid UAVs are not the most challenging UAV in autonomous flying. The flapping wing UAV, a kind of a broader class, called morphing UAVs, imposes real challenges and difficulties in its control. The morphing UAV is inspired by the idea of birds changing their shapes and the orientation of their wings to reach the most efficient way of flying according to the flying modes, such as flying speed and heading, and to the flying conditions such as weather and winds. This system's operation depends not only on the innovative materials and actuators incorporated into the body design but also on the intelligent control system. Therefore, such a system's autonomous control is so delicate and requires an exact computation to determine the best flight mode under the current conditions, adjust it continuously, and command the motors to execute the maneuver based on the position or/and velocity trajectory profiles.

Researches efforts have been shown in [31] in providing a deep reinforcement learning-based control system for that UAV. Because of the difficulties of the control situation, continuity in action and state spaces is needed, making the model-free Deep Deterministic Policy Gradient (DDPG) algorithm a good candidate for this task actor-critic policy. In categorizing the control into inner and outer loops, the deformation control is a part of inner loop control, and it is achieved by controlling the steering gears fixed on the airfoil. This algorithm is policy-based because it selects the next action based on the current state of the system. Under the umbrella of policies, it is worth mentioning that actors and critics have different policies; the former is a random policy made mainly for exploration. The latter is a deterministic policy made mainly for the evaluation of actions based on the defined reward function. The reward function is based on the angular orientation as well as linear position tracking.

The field of reinforcement learning is as fast-developing as innovative methods and improvements to the current methods. This long quest for high precision in achieving the control task and higher complexity in the tasks themselves will always bring miracles to the fast-evolving control field. Accordingly, a famous DRL method, which is DDPG, was manipulated and improved in research, shown in [64], to make a robust deep deterministic policy gradient (Robust-DDPG). The newly proposed method had shown better performances than the well-known algorithm for DDGP in terms of convergence rate, stability, and convergence efficiency of the UAV. The improvements to the control model were achieved using three modifications. The first improvement was the "delayed learning trick." It is achieved by delaying the update of the actor and critic networks. Rather than being updated after each iteration, both networks will be updated after each episode finishes. This improvement is to have stable learning under dynamic changes in the uncertain environment chosen for this task. The second improvement was the "adversarial attack trick." It is achieved by filtering the data used in the learning process by removing noisy states and actions from the sample. The reason behind this improvement is to have robust control in an uncertain environment. The third and last improvement was the "mixed exploration trick." It is achieved by a fine sampling of learning data based on Gaussian and rough sampling of that data based on a greedy approach. This fine-tuning of the model enables a high-speed convergence rate. The control variables for this model are the roll angle and the linear speed. Again, the reward function is formulated so that the goal is achieved for this particular task. Thus, the reward discussed in [64] depends on choosing the shortest path to the goal, approaching the goal with high speed, approaching the target in all movement rather than going away finally avoiding collisions well before encountering or hitting them.

Here, questions arise about the nature of the data on which the learning procedure in the deep reinforcement learning is done. Can the control system based on deep reinforce-

ment learning learn from images? If yes, should the images have high quality to be helpful for the learning process? These questions, and many more issues, are tackled in [65,66].

The control used in [66] aimed to achieve a precise autonomous driving of a UAV while performing landing on a platform. The landing area is marked with special visual markers, and perception through camera images is used in this process. The images can have low quality, and the DRL algorithm used in this control process is Deep Q-Learning Networks (DQNs). This UAV's mission is divided into two subtasks: detection of the marks indicating the landing area and execution of the vertical landing. Each task has its independently-designed DQN. However, these networks can have internal communication related to the mission. After analysis, the researcher has used double DQN architecture better to fit the system's nature and complexity. The process suffered from unreliable rewards due to Markov Decision Processes' dependence, which makes the reward sparse. To solve this problem, "partitioned buffer replay" is used, which is a method used in categorizing the experience data to facilitate the decision-making process.

Although the Deep Q-Networks method looks promising in solving control problems related to driving the unmanned aerial vehicle, several limitations prevent it from being applied in more advanced vehicle control tasks depending on image processing. As proposed in [65], deep reinforcement learning was used in controlling a UAV with the mission of capturing images of the front view of a person, especially his face. The drone should adjust its height and position autonomously to capture high-quality images using the onboard camera. The desired control was designed to be continuous, making the DQNs method unsatisfactory in usage because it is not suitable or robust for continuous problems with significant learning and testing datasets of raw pixel data. The DRL was used to enable performance and face tracking even under low-resolution input images due to environmental conditions. Under that scenario, Deep Deterministic Policy Gradient (DDPG) was applied with an actor-critic framework. Experimentation was done for the selected algorithm, and the results showed the superiority of such technique to classical control methods, represented in proportional-integral-derivative (PID) control and well-known 2D face recognition algorithms. The reward selection plays a crucial role in the speed of learning and applying and the accuracy of the tracking process. Accordingly, the reward was selected based on the control error in achieving a precise positioning to take the facial shots. Finally, the proposed method in [65] has proved to work well even under discrete control commands, like the one in the simulation environment.

After discussing several methods used for controlling the vehicle to a desired position and orientation to perform a specific task, larger missions and more challenging tasks are also used in the control of UAVs. More precisely, the outer loop control, which includes vehicle-to-vehicle communications, is a crucial topic in UAVs' real applications. Given the complexity of the task, the high speed of operation needed, and the precision that should be guaranteed, deep reinforcement learning is used for controlling a swarm of drones. Several approaches are used, with research still being ongoing in this area.

For example, in [67], the deep convolutional neural network control policy (DCNNP) is used to control five UAVs. The swarm's missions were to arrange themselves in a specific shape autonomously and defend an area from attackers. Driven by the necessity of cooperative behavior, Multi-Agent Reinforcement Learning (MARL) was used to solve the lack of information about all state variables for all UAVs in the swarm. In this control problem, centralized learning and centralized execution approaches were used in the DCNNP algorithm. This approach permits only a single UAV, named as an agent, to learn the model, plan behaviors for all the other UAVs, and communicate with them about their tasks. The system has a single output, as inherited from the CNN algorithm. The behavior of the DCNNP control was compared with the behavior of random policy, represented by moving the UAVs randomly untill achieving the mission, and of perfect/perimeter policy, represented in moving the UAVs equally and in the same direction. The results showed that the DCNNP vastly outperformed the random policy in both tasks; slight differences in performance when compared to the perimeter approach in the base defense were observed,

but with a reduced performance if compared to the perfect policy, especially when the number of UAVs is small in surveillance tasks.

Another research related to swarm control was proposed in [68], where a novel algorithm was developed, and it is called Deep Reinforcement Learning based Energy Efficient Control for Coverage Connectivity ($DRL - EC^3$). It is based on the Deep Deterministic Policy Gradient (DDPG) method with some modifications and the actor-critic policy using a two-layer deep neural network for each of them. Tensorflow is used to implement this algorithm, with 400 neurons in the first layer and 300 in the second one, in each network. This research's main objective was to decrease the energy consumption of the swarm UAVs during operation and communication without reducing the accuracy in task execution. The reward used to help in the learning was based on coverage, fairness index of coverage, energy consumption, and continuous connectivity in communication for all UAVs. This study showed that the proposed $DRL - EC^3$ control model outperforms both the well-known random and greedy policies. Moreover, the study showed that when the number of UAVs increased, the new control algorithm succeeded.

Still concerned with swarm control, the research proposed in [69] tackles the control problem with a new approach. This approach is the channel allocation for UAVs to maintain communication about the control tasks, such as state estimation and trajectory follow-up. Such a task involves an enormous amount of data from training and learning and real-time processing of the data. Thus, the Q learning was excluded from the options due to its difficulties in handling large datasets. Instead, a deep reinforcement learning network was used due to its capability to represent the data at a lower dimension and get the Q value more efficiently. The control architecture can be described as a multi-agent one with decentralized control, while no models for the system are needed, and the behavior of the other agents is viewed from the perspective of each agent as state observation for the surrounding. The deep reinforcement learning algorithm deals with four cases of channel status. First, if the channel is occupied, the system goes waiting until data can be sent again. Second, if the transmission fails, data must be re-sent. Third, if the transmission succeeds, the subsequent tasks can be performed. Fourth, if there is no task, wait idly. Because the main controller's aim lies in the quality of transmission, the reward is based on maximum data transmitted and lower transmission delay. The proposed strategy succeeds in providing accurate and fast operational behavior for communication between UAVs.

Accordingly, deep reinforcement learning (DRL) made a real scientific revolution in UAV control from different points of view. None can deny that the higher accuracy of the controller behavior motivates researchers to go the extra mile in using these advanced algorithms that require more computational time than classical control. Not to mention, deep reinforcement learning allows the researcher to tackle new missions that were nearly impossible before. The fact that the development of new DRL methods and strategies is still an ongoing process with many capabilities is promising news for research in UAV control.

6. Analysis and Insights

The challenging problem of real-world application of drones, i.e., package delivery, cave exploration, and mapping, needs to deal with uncertainties and be adaptive to the dynamic unknown environments. DRL offers a flexible framework to solve these problems without a model and can learn from experience and predict the correct solution for future observations. There are two branches of RL algorithms: model-based and model-free; model-based RL algorithms try to choose the optimal policy based on the learned model of the environment, while in model-free algorithms, the optimal policy is chosen based on the trail-and-error experienced by the agent. Both model-free and model-based algorithms have their upsides and downsides, but here we listed the Table 1 summary of model-free RL algorithms.

The decision-making function (control strategy) of the agent, which represents a mapping from situations to actions, is called policy, and there are two types of policy, i.e., on-policy and off-policy. On-policy methods attempt to evaluate or improve the policy

used to make decisions, whereas off-policy methods evaluate or improve a policy different from that used to generate the data. To illustrate the performance of on-policy (i.e., sarsa, PPO, TRPO) and off-policy (i.e., Q-learning, DQN, DDPG) RL, a random simulation is sampled, and the step response for each attitude command is displayed in Figure 7 [70], along with the target angular velocity to achieve Ω^*.

All algorithms reach some steady state; however, TRPO and DDPG have extreme oscillations in both the roll and yaw axis, which would cause instability during flight. However, globally speaking, in terms of error, PPO has shown to be a more accurate attitude controller. This is why, recently, Davide Scaramuzza et al. [71] train the drone using the PPO algorithm for the drone racing tracks AlphaPilot and Airsim drone because of its first-order policy gradient method that is particularly popular on its excellent benchmark performances and simplicity in implementation.

For training the drone's flight control tasks, all the algorithms of DRL have their upsides and downsides, we ran experiments using PPO, DDPG, and TRPO, on the hover, land, random waypoints, and target following. The comparative average reward results for these tasks using PPO, DDPG, and TRPO are shown in Figure 8. We found that we were typically able to solve the tasks within around 5000 policy iterations, with a batch size of 256 timesteps. We found that PPO was generally the most consistent algorithm for landing, followed by PPO and TRPO. For hover, interestingly, DDPG produced very smooth flight performance in comparison to the other algorithms. In the random waypoint task, PPO produced the smoothest waypoint navigation in comparison to the other algorithms. For target following task, DDPG produces better average rewards and smoothest waypoint navigation, followed by PPO and TRPO as shown in Figure 8 and the taxonomy of deep RL algorithms for UAVs tasks in Figure 3.

Table 1. Table summary of model-free RL algorithms.

Algorithm	Agent Type	Policy	Policy Type	MC or TD	Action Space	State Space
State action reward state action (SARSA) SARSA Lambda	Value-based	On-policy	Pseudo-deterministic ($\epsilon - greedy$)	TD	Discrete only	Discrete only
Deep Q Network (DQN) Double DQN Noisy DQN Prioritized Replay DQN Dueling DQN Categorical DQN Disturbuted DQN (C51)	Value-based	Off-policy	Pseudo-deterministic ($\epsilon - greedy$)		Discrete only	Discrete or Continuous
Normalized Advantage Functions (NAF) = Continuous DQN	Value-based				Continuous	Continuous
REINFORCE (Vanilla policy gradient)	Policy-based	On-policy	Stochastic	MC		
Policy Gradient	Policy-based		Stochastic			
TRPO	Actor-critic	On-policy	Stochastic		Discrete or Continuous	Discrete or Continuous
PPO	Actor-critic	On-policy	Stochastic		Discrete or Continuous	Discrete or Continuous
A2C/A3C	Actor-critic	On-polciy	Stochastic	TD	Discrete or Continuous	Discrete or Continuous
DDPG	Actor-critic	Off-policy	Deterministic		Continuous	Discrete or Continuous
TD3	Actor-critic				Continuous	Discrete or Continuous
SAC	Actor-critic	Off-policy			Continuous	Discrete or Continuous
ACER	Actor-critic				Discrete	Discrete or Continuous
ACKTR	Actor-critic				Discrete or Continuous	Discrete or Continuous

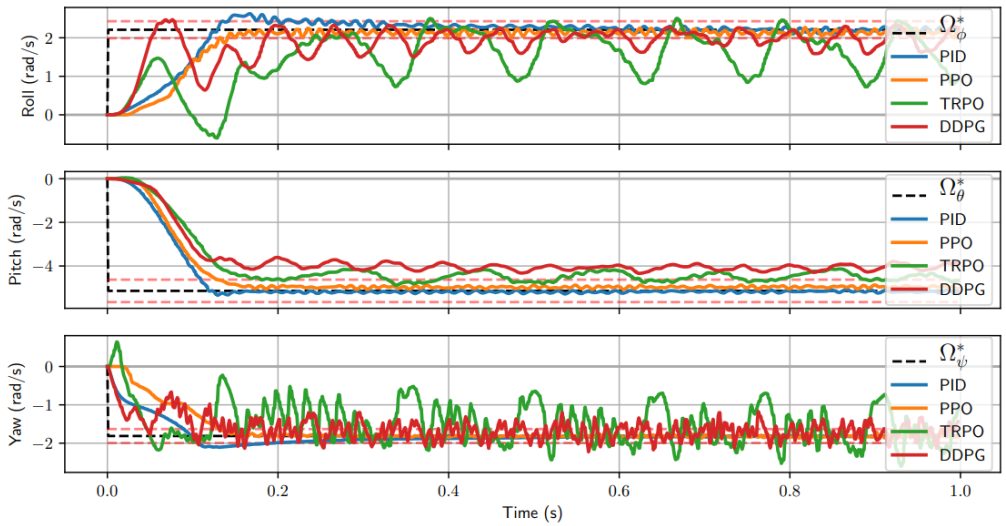

Figure 7. Step response of best trained RL agents compared to PID. Target angular velocity is $\Omega^* = [2.20, -5.14, -1.81]$ rad/s shown by dashed black line [70].

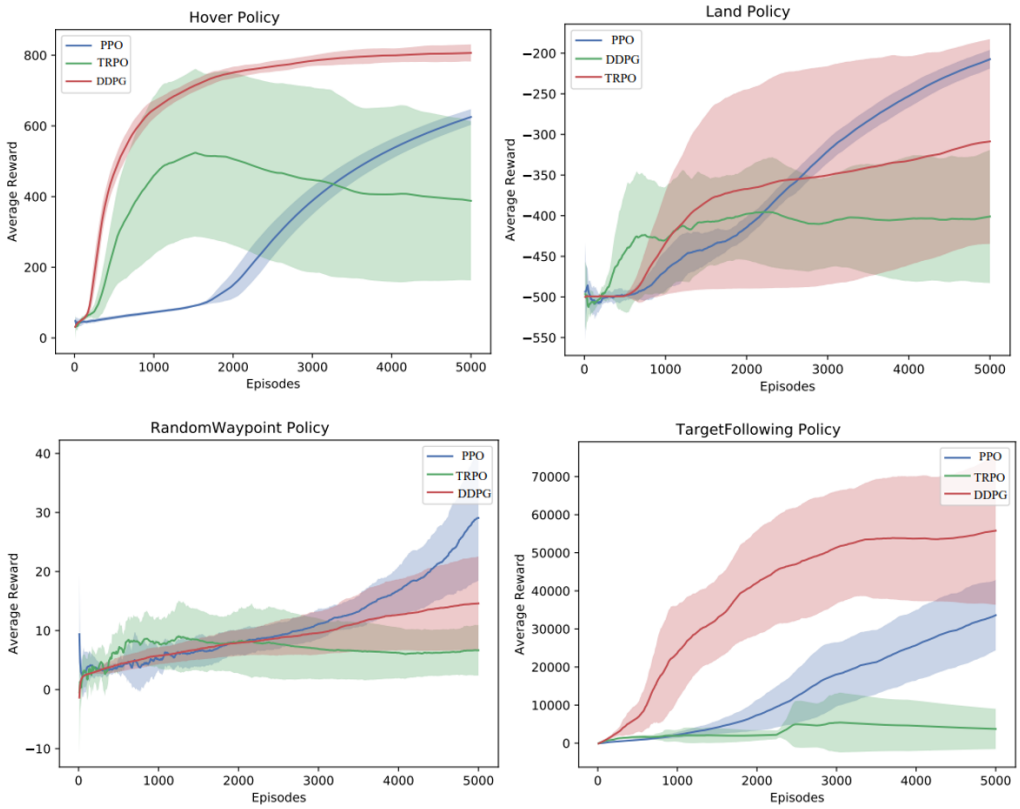

Figure 8. Average reward for the hover, land, random waypoint and target following tasks over 5000 iterations.

Inspired by the variations of Deep Deterministic Policy Gradient or DDPG, the algorithm proposed in this paper [72], which uses a mixture of policy gradients, critic network, a delayed update of the actor-networks and adds noise to the target policy has predicted next actions. Using this algorithm, we trained a drone to take off from the ground and reach a given target position through navigating a complex environment. The DDPG agent used a prioritized experience replay buffer to give higher importance to actions that yielded a high reward change. The Ornstein–Uhlenbeck process was used to add noise to the drone's rotor speeds to encourage exploration as the DDPG agent was made up of an actor and two critics. The

actor would learn what actions to make to maximize its rewards given the drone's position, and the critic networks predicted the rewards, or Q value, that the agent would receive given a state-action pair. To improve stability in the learning process, target networks for the critic networks and actor-network were defined, and they were updated softly by applying a τ discount of 10^{-3}. The waypoint navigation of drones in 3D simulation using DDPG is shown in Figure 9. The simulation framework and codes are open-sourced for use by the community (https://github.com/Kazimbalti/Drone-Deep-Reinforcement-Learning-.git) (accessed on 20 April 2021).

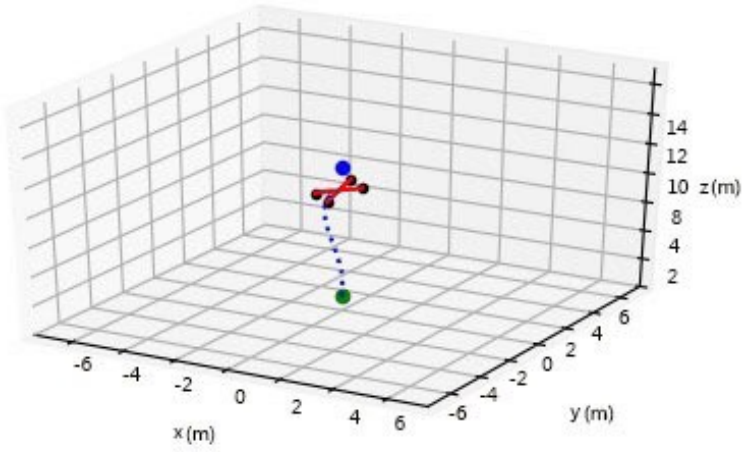

Figure 9. Waypoint navigation of drones using DDPG in 3D.

These findings suggest that deep RL has the potential for generating adaptive time-optimal trajectories for drones and merits further investigation.

7. Discussion

Regarding the usage of deep reinforcement learning in path planning, the first model proposed by [44] works well in indoor environments; however, it can be modified in the future to work for outdoor environments. This can be done by considering the height control of UAV in the model inputs. This enhancement will affect the feasible action space and other inputs. For instance, a solution for object avoidance could be achieved by only changing the UAV height without moving right or left.

The used DRL method learns the Q-values and the optimal policy for object avoidance. Other policies and methods can be used, such as Asynchronous Advantage Actor-Critic (A3C), deep deterministic policy gradient (DDPG), and dueling network architecture for double-deep Q-networks (D3QN). Only the loss function should be changed.

For large-scale complex unknown environments, some ideas go toward using DRL techniques to go forward to its mission, relying only on sensory data and GPS signals. It would be unnecessary to plan a vague 3D model for unknown environment and plan paths for vehicles to follow. The UAV will be able to navigate intelligently from arbitrary departure places to arbitrary target positions [73]. Moreover, it may go to a new unknown environment without a pre-planned back trip after the mission.

DRL use becomes fruitful because it should be pushed to simplify many other tasks for UAVs. One of such implementations is the landing scenario for the vehicle. Normally, it relies on ground cameras, range sensors, GPS, and more to land precisely in the end position, avoiding any failure [66]. With deep reinforcement learning, a rising idea discusses the combination of it and only a downward-looking camera to land precisely. The DRL may support two main stages of marking the end position to be detected and the vehicle's vertical descending. Those networks' training will be most helpful for UAVs and executed in a faster manner [66].

Regarding overall systems enhancement, environmental conditions such as wind speed, rains, and dust should be considered to perform a type of uncertainty in results. So, it should be included as system disturbances and then handled professionally.

Future work is not restricted to system development and enhancement approaching higher precision, accuracy, efficiency, and save energy; However, a part of researches and implementation should focus on implementing the current state of knowledge using reinforcement learning on different important applications such as wildfire monitoring, rescue missions, and dangerous environments exploration [34].

UAV navigation and control have been an emerging field due to its wide range of applications from another point of view. UAVs are being used in different aspects of life, such as civilian tasks, military missions, object tracking, and search and rescue missions [74]. Each of those applications requires precise navigation to avoid collisions while taking the optimum route to reach the target. Navigation through an unknown environment with changing obstacles is one of the challenges that face the UAVs [75]. In order to address this challenge, various control algorithms and AI techniques were developed and tested.

Based on the various trials conducted using learning algorithms to help the UAV determine the optimum path in real-life, DRL is one of the promising solutions in UAV navigation. Other machine learning algorithms need labeled data to start the training process, which is not available in UAV cases as, in reality, the UAV deals with new environments during each mission [73]. Using RL in UAV navigation relies mainly on the reward function, which is determined by the UAV's actions. One of the algorithms used in training RL is the Q-network, which can be combined with different neural networks to determine the optimal action value. The algorithm relies on the continuous updating for the UAV states based on the data generated from onboard sensors, then determining the optimum action to be taken and the associate reward value.

RL algorithm can also be tuned and trained using a PID controller. The controller uses the inputs and the observation of the current state of the UAV to update the reward value and the next action accordingly. The PID controller takes the inputs as the data from the onboard sensors and the values of the three gains by which the system's stability is determined. Several studies focused on using UAVs in SAR missions by applying RL algorithms to achieve optimum navigation. Different function approximation algorithms such as FSR were used with RL to ensure the convergence of the Q-table while representing the current state of the UAV. As autonomous navigation is one of the emerging fields, several mentioned studies focus on simulated experiments to apply the learning algorithms. Future studies will explore the real-life implementation of more advanced learning algorithms to ensure stable and smooth autonomous navigation for UAVs, along with exploring the capabilities of new RL algorithms in drone navigation.

Additionally, the usage of deep reinforcement learning in UAV lateral and longitudinal control and communication between swarms of UAVs and the organization of tasks marks an important progressive step in the possible applications for UAVs and the precision obtained. To sum up, in [29,37,63], the PPO algorithm was proved to provide a precise and fast performance in controlling the attitude of UAVs of different types; fixed-wing UAV, hybrid UAV, and quadrotor, respectively. Moreover, in controlling the flapping wing UAV in [31], the DDPG algorithm was used to control the flapping behavior. On the other hand, the research work is shown in [64] was based on making improvements for the DDGP algorithm represented in making it robust for better handling uncertainties and disturbances.

Then, in tackling multiple UAV control problems, deep reinforcement learning was used to control a fleet of UAVs while performing specific tasks with minimum movements and elapsed time. Ref. [67], DCNNP algorithm was used as a means for control, while in the [68], a new algorithm called $DRL - EC^3$ was used in the control process for energy efficiency.

The researches in the literature in the field of UAV control are fascinating, not only because many of them featured improvements to existing DRL algorithms or even propos-

ing new ones, but also because they are applied to vast areas of applications and were used to handle control problems that were nearly impossible to do before. In [65,66], the learning process in the DRL algorithm was based on information extracted from images. Thus, the advances in single and multi UAVs control using different communication and perception methods open the doors for vast real implementation of these mechanisms in many tasks like surveillance, first responders in catastrophes, transportation, agriculture, etc. It is apparent in all these researches and many more that the choice of reward function is as important as the DRL algorithm choice. Each research shows a new reward function based on the research objective and the mission of the control algorithm. This means that a thorough analysis of reward functions is needed, and a trial of different reward functions under the same control algorithm can be promising and lead to higher efficiency control implementation.

Future work in UAV control should be concerned with more experimental trials of the control methods, especially in the open air under uncertainties and unpredicted disturbances, to better test the models' validity. Moreover, research towards energy-efficient control should be addressed in more research, especially for quadrotors. After all, currently, their flight time is greatly limited because they consume a lot of energy. In the area of control of multiple UAVs, the idea of cooperation of multiple UAVs with one or more mobile robots for better exploration of sites should be researched well, and the appropriate control algorithms for such cooperation should be tested validated. For example, compliance with air space regulations, separation management in case of multiple drones, route planning and rerouting, sequencing and spacing, dynamic geofencing, terrain avoidance, contingency management, congestion management, and severe-weather redirection are identified by McKinsey in [76] as challenging aspects of using drones in last-mile delivery services. McKinsey highlighted the need for sophisticated unmanned-traffic-management (UTM) systems that properly control and manage drone flights.

In a nutshell, deep reinforcement learning in path planning, navigation, and control of UAVs has made revolutions in the field. Fortunately, improvements to the DRL control algorithms and the mechanical design of UAVs are continuously made and tested. Thus, this marks the birth of new challenging missions and applications for different categories of UAVs.

8. Conclusions

UAVs have been categorized according to various aspects such as top-level configuration, altitude, take-off weight, level of autonomy, and ownership. UAV's autonomy addresses a UAV's ability to take off, execute a mission, and return it to its base without any significant human intervention. The autonomy level is crucial for the successful deployment of UAVs in various challenging application domains and under various harsh environmental conditions. A UAV can be considered fully autonomous if it can make decisions and react to unexpected events in real-time without humans' direct intervention. To achieve this level of autonomy, many technological and algorithmic developments are still needed. Recently, there have been attempts to incorporate Artificial intelligence techniques such as DRL to safely enable a UAV to navigate in an unknown environment. In this review paper, deep reinforcement learning techniques for path planning, navigation, and UAVs control are reviewed. More experimental trials are required in an outdoor environment and under various environmental conditions.

Author Contributions: Conceptualization, A.T.A., A.K., methodology, A.T.A., A.K., N.A.M., H.A.I., Z.F.I., M.K., A.A., B.B., A.M. K.,I.A.H. and G.C.; software, M.K., A.M.K. and A.A.; validation, A.T.A., A.K., A.M.K., I.A.H., G.C.; formal analysis, I.K.I., A.T.A.; Investigation, N.A.M., H.A.I., Z.F.I., A.A., B.B., A.M.K., I.A.H. and G.C.; resources, N.A.M., H.A.I., Z.F. I., M.K., A.A., B.B., A.M.K., I.A.H. and G.C.; Data Curation, A.M.K., B.B., I.A.H., G.C.; writing—original draft preparation A.T.A., A.K., N.A.M., H.A.I., Z.F.I., M.K.; writing—review and editing A.T.A., A.K., N.A.M., H.A.I., Z.F.I., M.K., A.A., B.B., A.M.K., I.A.H. and G.C.; visualization, N.A.M., H.A.I., Z.F.I., M.K., A.A., B.B., A.M.K.; Supervision, A.T.A., and A.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: We would like to show our gratitude to Prince Sultan University, Riyadh, Kingdom of Saudi Arabia. Special acknowledgement to Robotics and Internet-of-Things Lab (RIOTU), Prince Sultan University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Narayanan, R.G.L.; Ibe, O.C. Joint Network for Disaster Relief and Search and Rescue Network Operations. In *Wireless Public Safety Networks 1*; Elsevier: Amsterdam, The Netherlands, 2015; pp. 163–193.
2. Suli, F. *Electronic Enclosures, Housings and Packages*; Woodhead Publishing: Cambridge, UK, 2018.
3. Tsiatsis, V.; Karnouskos, S.; Holler, J.; Boyle, D.; Mulligan, C. *Internet of Things: Technologies and Applications for a New Age of Intelligence*; Academic Press: Cambridge, MA, USA, 2018.
4. Castellano, G.; Castiello, C.; Mencar, C.; Vessio, G. Crowd detection in aerial images using spatial graphs and fully-convolutional neural networks. *IEEE Access* **2020**, *8*, 64534–64544. [CrossRef]
5. Kim, I.; Shin, S.; Wu, J.; Kim, S.D.; Kim, C.G. Obstacle avoidance path planning for uav using reinforcement learning under simulated environment. In Proceedings of the IASER 3rd International Conference on Electronics, Electrical Engineering, Computer Science, Sapporo, Japan, 13–17 May 2017; pp. 34–36.
6. Custers, B. Drones Here, there and everywhere introduction and overview. In *The Future of Drone Use*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 3–20.
7. Samanta, S.; Mukherjee, A.; Ashour, A.S.; Dey, N.; Tavares, J.M.R.S.; Abdessalem, K.W.; Taiar, R.; Azar, A.T.; Hassanien, A.E. Log Transform Based Optimal Image Enhancement Using Firefly Algorithm for Autonomous Mini Unmanned Aerial Vehicle: An Application of Aerial Photography. *Int. J. Image Graph.* **2018**, *18*, 1850019. [CrossRef]
8. Najm, A.A.; Ibraheem, I.K.; Azar, A.T.; Humaidi, A.J. Genetic Optimization-Based Consensus Control of Multi-Agent 6-DoF UAV System. *Sensors* **2020**, *20*, 3576. [CrossRef] [PubMed]
9. Azar, A.T.; Serrano, F.E. ; Kamal, N.A.; Koubaa, A. Leader-Follower Control of Unmanned Aerial Vehicles with State Dependent Switching. In *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 862–872.
10. Azar, A.T.; Serrano, F. E. ; Kamal, N.A.; Koubaa, A. Robust Kinematic Control of Unmanned Aerial Vehicles with Non-holonomic Constraints. In *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 839–850.
11. Azar, A.T.; Serrano, F.E. ; Koubaa, A.; Kamal, N.A. Backstepping H-Infinity Control of Unmanned Aerial Vehicles with Time Varying Disturbances. In Proceedings of the 2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH), Riyadh, Saudi Arabia, 15–17 March 2020; pp. 243–248.
12. Dalamagkidis, K. Definitions and terminology. In *Handbook of Unmanned Aerial Vehicles*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 43–55.
13. Valavanis, K.P.; Vachtsevanos, G.J. *Handbook of Unmanned Aerial Vehicles*; Springer: Berlin/Heidelberg, Germany, 2015.
14. Dalamagkidis, K.; Valavanis, K.P.; Piegl, L.A. *On Integrating Unmanned Aircraft Systems into the National Airspace System: Issues, Challenges, Operational Restrictions, Certification, and Recommendations*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011; Volume 54.
15. Weibel, R.; Hansman, R.J. Safety considerations for operation of different classes of UAVs in the NAS. In Proceedings of the AIAA 4th Aviation Technology, Integration and Operations (ATIO) Forum, Chicago, IL, USA, 20–22 September 2004; p. 6244.
16. Huang, H.M. Autonomy levels for unmanned systems (ALFUS) framework: Safety and application issues. In Proceedings of the 2007 Workshop on Performance Metrics for Intelligent Systems, Washington, DC, USA, 11–13 October 2007; pp. 48–53.
17. Clough, B.T. Unmanned aerial vehicles: Autonomous control challenges, a researcher's perspective. In *Cooperative Control and Optimization*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 35–52.
18. Protti, M.; Barzan, R. *UAV Autonomy-Which Level Is Desirable?-Which Level Is Acceptable? Alenia Aeronautica Viewpoint*; Technical Report; Alenia Aeronautica SPA Torino: Torinese, Italy, 2007.
19. Tüllmann, R.; Arbinger, C.; Baskcomb, S.; Berdermann, J.; Fiedler, H.; Klock, E.; Schildknecht, T. On the Implementation of a European Space Traffic Management System-I. A White Paper. 2017 Available online: https://www.semanticscholar.org/paper/On-the-Implementation-of-a-European-Space-Traffic-A-Tuellmann-Arbinger/6ac686ded55171072aa719c7c383e55c3cd059e2 (accessed on 5 January 2021)
20. Arulkumaran, K.; Deisenroth, M.P.; Brundage, M.; Bharath, A.A. Deep reinforcement learning: A brief survey. *IEEE Signal Process. Mag.* **2017**, *34*, 26–38. [CrossRef]
21. Poole, D.L.; Mackworth, A.K. *Artificial Intelligence: Foundations of Computational Agents*; Cambridge University Press: Cambridge, UK, 2010.
22. François-Lavet, V.; Henderson, P.; Islam, R.; Bellemare, M.G.; Pineau, J. An introduction to deep reinforcement learning. *Found. Trends Mach. Learn.* **2018**, *11*, 219–354. [CrossRef]

23. Zhang, H.; Yu, T. Taxonomy of Reinforcement Learning Algorithms. In *Deep Reinforcement Learning*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 125–133.
24. Huang, H.; Yang, Y.; Wang, H.; Ding, Z.; Sari, H.; Adachi, F. Deep reinforcement learning for UAV navigation through massive MIMO technique. *IEEE Trans. Veh. Technol.* **2019**. [CrossRef]
25. Cao, W.; Huang, X.; Shu, F. Unmanned rescue vehicle navigation with fused DQN algorithm. In Proceedings of the 2019 International Conference on Robotics, Intelligent Control and Artificial Intelligence, Shenyang, China, 8–11 August 2019; pp. 556–561.
26. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
27. Shin, S.Y.; Kang, Y.W.; Kim, Y.G. Automatic Drone Navigation in Realistic 3D Landscapes using Deep Reinforcement Learning. In Proceedings of the 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), Paris, France, 23–26 April 2019; pp. 1072–1077.
28. Wang, Z.; Schaul, T.; Hessel, M.; Van Hasselt, H.; Lanctot, M.; De Freitas, N. Dueling network architectures for deep reinforcement learning. *arXiv* **2015**, arXiv:1511.06581.
29. Bøhn, E.; Coates, E.M.; Moe, S.; Johansen, T.A. Deep reinforcement learning attitude control of fixed-wing uavs using proximal policy optimization. In Proceedings of the 2019 International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 11–14 June 2019; pp. 523–533.
30. Guo, S.; Zhang, X.; Zheng, Y.; Du, Y. An autonomous path planning model for unmanned ships based on deep reinforcement learning. *Sensors* **2020**, *20*, 426. [CrossRef]
31. Xu, D.; Hui, Z.; Liu, Y.; Chen, G. Morphing control of a new bionic morphing UAV with deep reinforcement learning. *Aerosp. Sci. Technol.* **2019**, *92*, 232–243. [CrossRef]
32. Lee, S.; Bang, H. Automatic Gain Tuning Method of a Quad-Rotor Geometric Attitude Controller Using A3C. *Int. J. Aeronaut. Space Sci.* **2019**, *21*, 469–478. [CrossRef]
33. Hardin, P.J.; Jensen, R.R. Small-scale unmanned aerial vehicles in environmental remote sensing: Challenges and opportunities. *GIScience Remote Sens.* **2011**, *48*, 99–111. [CrossRef]
34. Pham, H.X.; La, H.M.; Feil-Seifer, D.; Nguyen, L.V. Autonomous uav navigation using reinforcement learning. *arXiv* **2018**, arXiv:1801.05086.
35. Lin, Y.; Wang, M.; Zhou, X.; Ding, G.; Mao, S. Dynamic spectrum interaction of UAV flight formation communication with priority: A deep reinforcement learning approach. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *6*, 892–903. [CrossRef]
36. Li, B.; Wu, Y. Path planning for UAV ground target tracking via deep reinforcement learning. *IEEE Access* **2020**, *8*, 29064–29074. [CrossRef]
37. Koch, W.; Mancuso, R.; West, R.; Bestavros, A. Reinforcement learning for UAV attitude control. *ACM Trans. Cyber Phys. Syst.* **2019**, *3*, 1–21. [CrossRef]
38. Dhargupta, S.; Ghosh, M.; Mirjalili, S.; Sarkar, R. Selective opposition based grey wolf optimization. *Expert Syst. Appl.* **2020**, *151*, 113389. [CrossRef]
39. Qu, C.; Gai, W.; Zhong, M.; Zhang, J. A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning. *Appl. Soft Comput.* **2020**, *89*, 106099. [CrossRef]
40. Jiang, S.; Jiang, C.; Jiang, W. Efficient structure from motion for large-scale UAV images: A review and a comparison of SfM tools. *ISPRS J. Photogramm. Remote Sens.* **2020**, *167*, 230–251. [CrossRef]
41. He, L.; Aouf, N.; Whidborne, J.F.; Song, B. Deep reinforcement learning based local planner for UAV obstacle avoidance using demonstration data. *arXiv* **2020**, arXiv:2008.02521.
42. Bayerlein, H.; Theile, M.; Caccamo, M.; Gesbert, D. UAV path planning for wireless data harvesting: A deep reinforcement learning approach. *arXiv* **2020**, arXiv:2007.00544.
43. Hasheminasab, S.M.; Zhou, T.; Habib, A. GNSS/INS-Assisted structure from motion strategies for UAV-Based imagery over mechanized agricultural fields. *Remote Sens.* **2020**, *12*, 351. [CrossRef]
44. Singla, A.; Padakandla, S.; Bhatnagar, S. Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge. *IEEE Trans. Intell. Transp. Syst.* **2019**. [CrossRef]
45. Bouhamed, O.; Ghazzai, H.; Besbes, H.; Massoud, Y. Autonomous UAV navigation: A DDPG-based deep reinforcement learning approach. In Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 10–21 October 2020; pp. 1–5.
46. Challita, U.; Saad, W.; Bettstetter, C. Interference management for cellular-connected UAVs: A deep reinforcement learning approach. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 2125–2140. [CrossRef]
47. Yan, C.; Xiang, X.; Wang, C. Towards Real-Time Path Planning through Deep Reinforcement Learning for a UAV in Dynamic Environments. *J. Intell. Robot. Syst.* **2019**, *98*, 297–309. [CrossRef]
48. Wang, Y.M.; Peng, D.L. A simulation platform of multi-sensor multi-target track system based on STAGE. In Proceedings of the 2010 8th World Congress on Intelligent Control and Automation, Jinan, China, 6–9 July 2010; pp. 6975–6978.
49. Shin, S.Y.; Kang, Y.W.; Kim, Y.G. Obstacle Avoidance Drone by Deep Reinforcement Learning and Its Racing with Human Pilot. *Appl. Sci.* **2019**, *9*, 5571. [CrossRef]
50. Muñoz, G.; Barrado, C.; Çetin, E.; Salami, E. Deep Reinforcement Learning for Drone Delivery. *Drones* **2019**, *3*, 72. [CrossRef]
51. Hii, M.S.Y.; Courtney, P.; Royall, P.G. An evaluation of the delivery of medicines using drones. *Drones* **2019**, *3*, 52. [CrossRef]

52. Pham, H.X.; La, H.M.; Feil-Seifer, D.; Van Nguyen, L. Reinforcement learning for autonomous uav navigation using function approximation. In Proceedings of the 2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), Philadelphia, PA, USA, 6–8 August 2018; pp. 1–6.
53. Kahn, G.; Villaflor, A.; Pong, V.; Abbeel, P.; Levine, S. Uncertainty-aware reinforcement learning for collision avoidance. *arXiv* **2017**, arXiv:1702.01182.
54. Altawy, R.; Youssef, A.M. Security, privacy, and safety aspects of civilian drones: A survey. *ACM Trans. Cyber Phys. Syst.* **2016**, *1*, 1–25. [CrossRef]
55. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef]
56. Bamburry, D. Drones: Designed for product delivery. *Des. Manag. Rev.* **2015**, *26*, 40–48.
57. Li, J.; Li, Y. Dynamic analysis and PID control for a quadrotor. In Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation, Beijing, China, 7–10 August 2011; pp. 573–578.
58. Liu, Y.; Nejat, G. Robotic urban search and rescue: A survey from the control perspective. *J. Intell. Robot. Syst.* **2013**, *72*, 147–165. [CrossRef]
59. Tomic, T.; Schmid, K.; Lutz, P.; Domel, A.; Kassecker, M.; Mair, E.; Grixa, I.L.; Ruess, F.; Suppa, M.; Burschka, D. Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue. *IEEE Robot. Autom. Mag.* **2012**, *19*, 46–56. [CrossRef]
60. McClelland, J.L.; McNaughton, B.L.; O'Reilly, R.C. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychol. Rev.* **1995**, *102*, 419. [CrossRef]
61. Sutton, R.; Barto, A.G. *Reinforcement Learningan Introduction*; MIT Press: Cambridge, MA, USA, 1998.
62. Tai, L.; Liu, M. A robot exploration strategy based on q-learning network. In Proceedings of the 2016 IEEE International Conference on Real-Time Computing and Robotics (RCAR), Angkor Wat, Cambodia, 6–10 June 2016; pp. 57–62.
63. Xu, J.; Du, T.; Foshey, M.; Li, B.; Zhu, B.; Schulz, A.; Matusik, W. Learning to fly: Computational controller design for hybrid UAVs with reinforcement learning. *ACM Trans. Graph. (TOG)* **2019**, *38*, 1–12. [CrossRef]
64. Wan, K.; Gao, X.; Hu, Z.; Wu, G. Robust Motion Control for UAV in Dynamic Uncertain Environments Using Deep Reinforcement Learning. *Remote Sens.* **2020**, *12*, 640. [CrossRef]
65. Passalis, N.; Tefas, A. Continuous drone control using deep reinforcement learning for frontal view person shooting. *Neural Comput. Appl.* **2019**, *32*, 4227–4238. [CrossRef]
66. Polvara, R.; Patacchiola, M.; Sharma, S.; Wan, J.; Manning, A.; Sutton, R.; Cangelosi, A. Toward end-to-end control for UAV autonomous landing via deep reinforcement learning. In Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 12–15 June 2018; pp. 115–123.
67. Tožička, J.; Szulyovszky, B.; de Chambrier, G.; Sarwal, V.; Wani, U.; Gribulis, M. Application of deep reinforcement learning to UAV fleet control. In Proceedings of the SAI Intelligent Systems Conference, London, UK, 5–6 September 2018; pp. 1169–1177.
68. Liu, C.H.; Chen, Z.; Tang, J.; Xu, J.; Piao, C. Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 2059–2070. [CrossRef]
69. Yang, J.; You, X.; Wu, G.; Hassan, M.M.; Almogren, A.; Guna, J. Application of reinforcement learning in UAV cluster task scheduling. *Future Gener. Comput. Syst.* **2019**, *95*, 140–148. [CrossRef]
70. Koch, W. Flight controller synthesis via deep reinforcement learning. *arXiv* **2019**, arXiv:1909.06493.
71. Song, Y.; Steinweg, M.; Kaufmann, E.; Scaramuzza, D. Autonomous Drone Racing with Deep Reinforcement Learning. *arXiv* **2021**, arXiv:2103.08624.
72. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the International Conference on Machine Learning, Jinan, China, 26–28 May 2018; pp. 1587–1596.
73. Wang, C.; Wang, J.; Zhang, X.; Zhang, X. Autonomous navigation of UAV in large-scale unknown complex environment with deep reinforcement learning. In Proceedings of the 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Montreal, QC, Canada, 14–16 November 2017; pp. 858–862.
74. Imanberdiyev, N.; Fu, C.; Kayacan, E.; Chen, I.M. Autonomous navigation of UAV by using real-time model-based reinforcement learning. In Proceedings of the 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), Phuket, Thailand, 13–15 November 2016; pp. 1–6.
75. Bou-Ammar, H.; Voos, H.; Ertel, W. Controller design for quadrotor uavs using reinforcement learning. In Proceedings of the 2010 IEEE International Conference on Control Applications, Yokohama, Japan, 8–10 September 2010; pp. 2130–2135.
76. Duvall, T.; Green, A.; Langstaff, M.; Miele, K. *Air-Mobility Solutions: What They'll Need to Take off*; Technical Report; McKinsey: New York, NY, USA, 2019.