"A Systematic Review of DRL-Based Approaches for Autonomous Drone Navigation: A review"

Yatish SJ[1] , Viji Vinod[2]

[1]*PhD Scholar , Dr.M.G.R Educational and Research Institute,Chennai,*

[2]*Computer Applications, Dr.M.G.R Educational and Research Institute,*

yatishsjphd@gmail.com., hod-mca@drmgrdu.ac.in

Abstract:

The large range of applications for autonomous drone navigation, including search and rescue, aerial surveillance, and package delivery, has made it a prominent research topic in recent years. In comparison to flying drones through challenging terrain, navigating robots is very simple. DeepReinforcementLearning (DRL) has received a lot of interest since it has the experience learning ability to perform challenging tasks with little knowledge. Navigation is a drone's main fundamental difficulty. Deep reinforcement learning (DRL), which enables the drone to learn from its experiences and develop its decision-making abilities, has emerged as a viable technology for autonomous drone navigation. The literature on DRL-based algorithms, architectures, and applications for drone navigation is reviewed in this article. Additionally, it identifies the gaps in the existing body of research and analyses the difficulties with DRL-based techniques. Future directions for this field of study are discussed as the review comes to a close. Overall, this systematic review indicates potential for additional study in this area and offers a thorough grasp of the current state-of-the-art in DRL-based techniques for autonomous drone navigation.

Index Terms— *DRL, Navigation, Obstacle Avoidance.*

1.Introduction:

UAVs have revolutionized many fields, including wildlife monitoring and conservation. With their capacity to take pictures and gather information from far-off and difficult-to-reach places. UAVs have developed into a useful tool for academics and environmentalists. However, the use of UAVs for autonomous navigation and data collection still presents many challenges. The primary purpose of navigation in a two- or three-dimensional environment is to find an ideal or less ideal path between two points while avoiding obstacles. Robust robot navigation systems are necessary for automated guided vehicles for the warehouse, delivery robots, and indoor service robots in their dynamic surroundings..

The commonly used method of drone navigation system is vision-based motion method which was focused on combining many relevant algorithms one such is SLAM, or simultaneous localization and mapping Fig.1 SLAM, is frequently used in this technique. The SLAM has the ability to create a map of an unfamiliar landscape, utilise this technique to determine its precise location, and then begin moving in that direction using a path planning module.

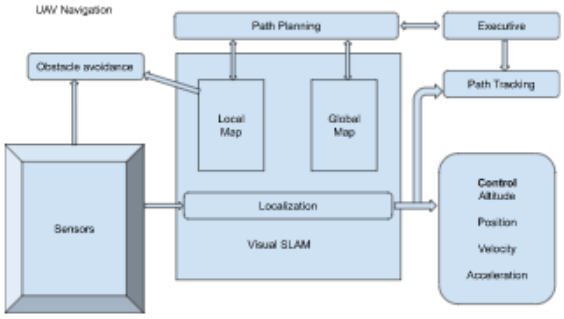

Figure 1. Common visual-based drone navigation framework.

The visual-based SLAM, position of the UAV and direction of the hovering drone is estimated by taking combined input from the onboard cameras, LiDAR, GPS and inertial data, after which complex mapping systems will be researched and improved for dynamic environments[1]. LSD-SLAM and ORB-SLAM are the two primary visual SLAM techniques.[2,4]. The two main challenges we face using the above algorithms : (1) developing a proper and effective image based on the data collected and (2).possiblities of wrong results in case of the object in an environment moves, changes in the camera parameter, switching between illumination. There are researches like the laser SLAM algorithm which right away builds a dense laser ranging method, such as GMapping[3] or Hector SLAM[5], which are used to develop an obstacle map of the environment.The drawbacks of laser SLAM includes (1). The time taken for building and updating the map is very high and (2). Due of the algorithm's dependence on accuracy of the sensors, there is a need for dense laser sensor.

Path Planning is another crucial module that may be used in the conventional drone navigation framework. The amount of environments information gathered from different sources, this module can be separated into global map and local map of path planning. Global map path planning deals with obtaining a complete route details depending on the recognised contexts. Recent studies have often employed techniques such the (A*) A-star[6], Artificial

bee colony optimisation[7], Optimization of Ant colony, and rapid-exploration random tree[8]. These techniques mostly rely on static maps, making it challenging to use them in a real-world setting. The issues caused by the regular changes in the environment and re-planning the local pathways are addressed by the implementation of local map path planning methods like Artificial Potential field (APF) and the technique using a dynamic window[9]. The issues encountered while putting classic path planning algorithms into practice are (1). While displaying a grid-based map, there are discrepancies between its accuracy and memory needs. (2). Real-time replanning of the navigation course in a real-world setting necessitates vast quantities of computations, which limits its responsiveness to some level.

The classic navigation framework's aforementioned components each provide a difficult research problem, and their implementation can occasionally result in significant computational mistakes. When these flaws are implemented in the real world, they perform poorly because they progressively accumulate.

Deep Learning(DRL) has a potential approach to resolving these issues in recent years. DRL is another machine learning approach, which basically utilises both DNN with RL algorithms to help operators learn effectively in complex environments.

Several studies have explored the use of DRL for autonomous drone navigation, with a particular focus on developing algorithms that can handle the complexity of real-world environments. For example, Y. Xue et al. (2021) suggested a DRL-based strategy for drone navigation that uses a recurrent neural network (RNN)[9] to process sensor data and generate control commands. The approach was tested in both simulated and real-world environments, demonstrating its effectiveness in obstacle avoidance and navigation tasks.

Similar to this, Amer, K. et al. (2020) created a DRL-based solution for drone navigation that makes use of a deep Q-network (DQN) to build a convolutional neural network(CNN) and a navigation policy[10] to take use of sensor input's characteristics. The technology performed better than conventional techniques when evaluated in a simulated setting.

Likewise KaiZhu and Tao Zhang (2021) did a evaluation of mobile navigation using DRL from 2016 to 2020 and presented a comprehensive and systematic differences between traditional robot navigation framework and DRL operated robot navigation. The approach was based on the secondary research which showed us the difference of the robot navigation Indoor, multi-robot, social etc[11].

Although, many research were focused on mobile navigation of drones or robots. In order to create decentralised regulations for a robot team to safely travel over uncharted complicated terrain while preserving connectivity[12], Juntong Lin et al. (2019) devised a DRL-based approach to the multi-robot navigation problem.

Then again, consveration of energy while maintaining the UAVs a float and also making sure all the onboard sensors are active was a challenging task. For example, Chi Harold Liu et al. (2018) proposed a DRL- based method for UAV control. The approach tested multiple simulations to demonstrate its effectiveness[13]

Other studies have explored the use of DRL for specific drone navigation tasks, such as collision avoidance and target tracking. A team of drones may be trained to avoid collisions while navigating in a complex environment using a multi-agent reinforcement learning(MARL) framework, as demonstrated by Chen et al's (2020) DRL-based strategy for collision-free navigation[14]. The approach was tested in a simulated environment, showing improved performance compared to traditional methods.

Moreover, several studies have explored the use of DRL for drone navigation in GPS-denied environments, where traditional navigation methods may not be effective. For example, Yalong Pi et. al. (2020) developed a DRL-based approach for drone navigation in GPS-denied environments that uses a DQN[15] to learn a policy for navigation based on visual cues. The approach was tested in a simulated environment, showing improved performance compared to traditional methods.

While the use of DRL for autonomous drone navigation has shown promising results, several challenges remain. For example, the high-dimensional input space of sensor data can make it difficult to train DRL algorithms effectively[16]. Additionally, the limited battery life of drones can constrain the amount of data that can be collected[13] and also making it challenging to train DRL algorithms from scratch.

Overall, the literature suggests that DRL-based approaches have the potential to significantly improve the performance of autonomous drone navigation. However, further research is needed to address the challenges associated with these approaches and to develop more effective algorithms for real-world applications.

Autonomous drone navigation has emerged as a popular research area with applications in various fields such as surveillance, search and rescue[17], and delivery services with collison avoidance[18]. Traditional methods for drone navigation require accurate sensors and complex algorithms that can handle the complexity of real-world environments. However, deep reinforcement learning (DRL) has shown promising results for autonomous drone navigation. DRL enables the drone to learn from its experiences and improve its decision-making skills, which can help it navigate through complex environments more efficiently. The State-of-the-art in DRL-based techniques for autonomous drone navigation is reviewed systematically in this work.

2.Deep reinforcement learning context

2.1.Preliminary

RL is a type of ML where an agent in an environment is simulated in such a way that it tries to make its own decisions that maximizes results. The actions performed by an agent in the simulated environment is monitored through actions and feedback as reward. It is the education that is intended to be taught to students, teaching the government to take action and the promotion of ideas necessary to get the best out of it. Agent-to-agent interaction process.

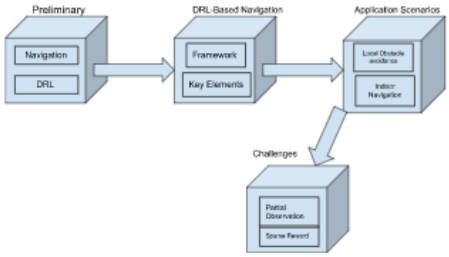

Figure 2. Flow-structure of this paper.

One type of ML called RL involves agents interacting within the simulated/real world to identify the best outcome to perform a particular task. Markov decision processes (MDP) can be used to quantitatively identify the mutual exchange between an agent and its simulated/real world.

MDPs model the environment as states, with the actions of agents for the transition of states. The agent always monitors the present state of the simulated/real-world, chooses the necessary action and receives a reward based upon the success of the action and the result of the change in the simulated/real-world state. The agent's goal is to analyze the rules that bind the current situation to action to maximize the opportunity.

The projected cumulative reward throughout the complete series of time steps serves as a proxy for the policy's effectiveness, which can be either stochastic or deterministic. By applying a discount factor to future benefits, one may calculate the predicted cumulative reward.

The agent generally employs an iterative method to develop an optimum policy, updating its strategy in response to incentives received and newly observed conditions. Value-based and policy-based algorithms[39,40] are the two primary categories of reinforcement learning algorithms.

The value function, which stands for the anticipated cumulative reward under a specific policy, is learned by value-based algorithms. The projected cumulative benefit of adopting a certain action in a specific state and subsequently continuing to follow the existing policy is represented by the Q-function, a particular kind of value function.

Without initially calculating the value function, policy-based algorithms simply learn the policy. To update the policy parameters, these algorithms frequently employ gradient descent techniques.

Let's define a Markov decision process (MDP) as a bundle (S, A, P, R, γ), where:

S is the possible process in the environment,

A is the process that does what the agent can do. do Take

is the probability of transition to state P(s's, a) ', i.e. the agent must act in state s

R(s, a, s') is the instant reward the agent receives when moving from state s' to state s', and γ is the discount rate that determines the future price of the soul of the product. γ is a value between 0 and 1; where 0 means only instant rewards are considered and 1 means all future rewards are considered significant.

At each step t, the agent monitors the current state s_t, chooses an action and receives a reward r_t. The agent's goal is to learn the law of π(as) that defines each case for the action that yields the greatest profit over time.

This is usually expressed as the reduced return value G_t:

$$G_t = r_t + 1 + \gamma\, r_t + 2 + \gamma^2 r_t + 3 + \ldots = \sum_{k=0}^{k=\infty} \gamma^k r_t + k + 1 \quad \sum_{k=0}^{k=\infty} \gamma^k r_t + k + 1 \quad (1)$$

The optimal rule π * is the rule that maximizes profit from each situation:

$$\pi^*(a/s) = argmax \sum_{k=0}^{k=\infty} \gamma^k r_t + k + 1 \quad \sum_{k=0}^{k=\infty} \gamma^k r_t + k + 1 \quad (2)$$

It is the responsibility of reinforcement learning algorithms to update the policy by responding to the rewards received and new situations found. Q-learning is a popular method that shows the Q-value function Q(s,a), which in some cases shows the expected profit from the best performance after taking a trade:

$$Q(s,a) = E[G_t / s_t=s, a_t=a, \pi^*] \quad (3)$$

The Q value function is modified using the Bellman equation, which establishes the relationship between the Q values of the subsequent pair of states and the state pair.:

$$Q(s,a) = E[r + \gamma max_{a'} Q(s',a') \mid s,a] \quad (4)$$

The broker updates its estimate of the Q-value function for each variable (s, a, r, s') using the update rule:

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma max_{a'} Q(s',a') - Q(s,a)) \quad (5)$$

Where α is the learning rate, which determines the rate at which the Q-values are updated.

Since real-time programming needs extensive real-time input and large memory utilisation, both of which are impracticable, academics have proposed and developed Monte Carlo and temporal-Difference(TD) learning as two alternative learning methodologies.

However, the classical RL method has serious problems in high speed situations because the need for computation increases the number of inputs. Therefore, deep learning ideas are put into practice.

Deep learning combines additional learning with deep neural networks and uses techniques such as graphs or raw data to learn new skills. To train neural networks to accurately anticipate the Q value function or rule, gradient descent is utilised, which lowers the prediction error. With a neural network, the agent decides what to do based on the current situation and adjusts the weights of the network according to the rewards received during training and new situations encountered.

Cost-based and policy-based approaches lead to two categories of DRLs.

Value-based deep reinforcement learning(DRL), a type of DRL algorithm, learns to provide a value to each state or state-function pair in a environment. The objective of cost-based economics is to identify the optimum strategy that maximises the projected return over time DRL[19].

In cost-based DRL, the broker learns an approximation of the cost function, expressed as Q-value (state-value) or V-value (state-value price). The Q-value of a state-action pair is the expected reward for initiating a state, performing an action, and then following a certain rule.

The V-value of a situation is the desired reward that will be obtained from a condition and then complete a rule.

In value-based DRL, the learning process must be iterated over the Q or V values using the Bellman equation, which is an iterative equation that connects the values of a state (or state-function pair) with values. her. neighbors (or pairs of states) and expected rewards from making a trade.

Mathematically, the Bellman equation for Q-values can be written as:

$$Q(s,a) = E[R(s,a)] + \gamma^* max(Q(s',a')) \quad (6)$$

When Q(s,a) is the Q value of the state-function pair (s,a), the expected reward from the state function R(s,a) and s' is the next state to be determined. A discount for the next Main reward, max(Q(s',a')) is the maximum Q of all potential actions in region s'.

The Bellman equation for V-values can be written as:

$$V(s) = E[R(s)] + \gamma^* max(V(s')) \quad (7)$$

where V(s) denotes the V-value of state s, R(s) is the anticipated reward for beginning in state s, and max(V(s')) denotes the highest possible V-value for all conceivable states in the next time step.

The Q-values or V-values are estimated by neural networks in value-based DRL algorithms like Deep Q-Networks (DQNs). The estimated Q-value (or V-value) for each action is produced by the neural network from the state (or state-action pair) input. (or state). To lessen the discrepancy between the goal Q value (or V value) and the estimated Q value (or V value), a type of stochastic gradient descent is employed to modify the neural network's parameters produced using Bellman equation.

To achieve the best predicted results in real time, deep learning (DRL) algorithms based on training policy directly improve the operational policy associated with the do. Policy-based DRL algorithms, as opposed to value-based DRL, learn to directly approximate the best policy without explicitly estimating the value function[20]. Value-based DRL determines how to value each state or state-action pair.

In a policy-based DRL system, the policy function might be represented by a neural network. The probability distribution across potential actions is output after accepting the situation as input. The goal of the policy-based DRL is to optimise the neural network's parameters over time to maximise the predicted cumulative reward. Gradient ascent on the predicted reward is often used to iteratively update the policy parameters in order to do this.

Mathematically, the objective of policy-based DRL can be written as:

$$\theta^* = argmax\ \theta\ E[R/\pi\theta(s,a)] \quad (8)$$

where R is the competitive reward, θ is the parameter of the neural network rule, and πθ(s,a) is the probability of action given the state s and the unconstrained rule θ. The objective function is usually optimized using gradient rise methods such as stochastic gradient rise to change the parameters.

are policy-based DRL algorithms that can be divided into two groups, Privacy Policy and Competitor Policy. Rule systems like REINFORCE use rewards from running instances of existing rules to update constraints to learn new rules. Deterministic policies such as Deep Deterministic Policy Gradints (DDPG) estimate the cost of a function to learn a policy and then use it to update the policy without restriction.

While the rule is optimized to generate a probability distribution between actions, rule-based DRL algorithms have the advantage of maintaining a fixed working space and stochastic space. They may be less useful for standards than cost-based systems because of the need to sample from existing policies.

2.2.DRL methods based on Value-Based

2.2.1.DQN:

To discover the best policy for a task, Thanh This Nguyen et al.(2020) developed in their review paper a deep learning method that blends Q-learning with deep neural networks[21].

Deep Q-Network (DQN) is a popular method for solving reinforcement learning (RL) problems using deep learning. The Q function calculates the future reward for each action for each environment predicted by DQN using a neural network. Learning the best way to maximize profits over time is the goal of DQN [22].

Iterative learning is a method used by the DQN algorithm to train neural networks. In the replay buffer, the controller saves the changes (s, a, r, s'); where s represents current state, a represents completion, r represents gifts received, and s' represents future state throughout the game.

Bellman's equation, a Q-learning algorithm version is used to train the neural network, which is updated by sampling a limited number of variables from the repeated parameters. According to the Bellman equation, a state-action pair's Q-value should be equal to the product of the immediate reward and the highest discounted Q-value, and the neural network is trained to reduce the discrepancy between them.

Mathematically, the update rule for the neural network in DQN is:

$$\theta \leftarrow \theta - \alpha \nabla Q(s,a;\theta)(Q(s,a;\theta) - (r + \gamma \max_{a'} Q(s',a';\theta^-))) \quad (9)$$

where θ is a weight set for the neural network, α is the learning rate, γ is the discount rate, Q(s, a; θ) is the estimated Q value for state s and action a, Q(s', a) '; θ^-) is the target Q value for the next state s' and each possible a', and r is the immediate reward. The main advantage of the DQN over traditional RL algorithms is that it can use deep neural networks to process states, thereby preserving complex patterns and relationships in the data.

2.2.2.Double Deep Q- Network:

A variant of DQN using two deep neural networks has been discussed to reduce overestimation of action[23]. The main idea of the

DDQN is to use two sets of Q networks, one for selecting actions (policy network) and one for evaluating actions (target network).Similar to the original DQN algorithm, the rule mesh is adjusted by minimizing the disparity between the anticipated and objective Q values. However, in DDQN the target Q-value is calculated using the target network instead of the network policy. The target mesh is periodically updated with the weights in the policy mesh, but these weights are kept constant during the calculation of the target Q-value.

This helps to avoid overestimating the Q value by reducing the noise in the policy network at the target Q value.

Mathematically, the update rule for the policy network in DDQN is:

$$L(\theta) = E[(r + \gamma\, Q(s', \operatorname{argmax}_{a'} Q'(s',a';\theta');\theta^-) - Q(s,a;\theta))^2] \quad (10)$$

where r is the reward, γ is the discount factor, Q' is the target Q-network, and Q is the policy Q-network. The argmax_a' operation is taken with respect to the policy network, but the Q-value for the selected action is evaluated using the target network.

The main advantage of DDQN over the original DQN algorithm is that it can significantly reduce the overestimation of Q-values, especially in high-dimensional action spaces. This can lead to more stable and accurate learning, as well as better performance on complex tasks[25].

2.3.DRL method based on Policy-Based

2.3.1.Proximal Policy Optimization (PPO):

A deep learning-based approach that directly improves bad policy by reducing the gap between current policy and previous policies has been discussed [6, 16].

The large variation of gradient estimates and the propensity to become trapped in local optima are two drawbacks of prior policy optimisation algorithms that PPO is intended to remedy. PPO does this by employing a substitute objective function that, while being simpler to optimise, approximates the goal of the policy optimisation problem.

The PPO method gathers batch of trajectories using the current policy repeatedly, uses a value function estimator to calculate the benefits of each action, and then modifies the policy depending on the gradient of the surrogate objective function. The clipped surrogate objective which guarantees that the ne policy is not too far from the previous policy, and an entropy term, which promotes exploration and prevents the policy from becoming overly

deterministic, are the two terms that make up the surrogate objective function.

Mathematically, the clipped surrogate objective is defined as:

$$L_{clip}(\theta) = E[min(r_t(\theta) * A_t, clip(r_t(\theta), 1-\varepsilon\varepsilon, 1+\varepsilon) * A_t)] \quad (11)$$

In this context, the rule parameter setting is represented by θ, while $r_t(\theta)$ refers to the ratio of the action probability under the new rule compared to the old rule. The effective value of the action over time t is represented by A_t, while the hyper-parameter check confidence size is represented by ε. Entropy, on the other hand, is defined as the negative probability of following the new rule divided by the hyper-parameter β, which controls the strength of the entropy constant.

The policy update is then performed by optimizing the surrogate objective using stochastic gradient descent or other optimization techniques.

PPO has been shown to be effective in a variety of challenging reinforcement learning problems, such as continuous control and robotic manipulation. It is also relatively easy to implement and tune compared to other policy optimization algorithms, making it a popular choice for researchers and practitioners in the field.

2.3.2.Asynchronous Advantage Actor-Critic (A3C):

Po-wei Chou(2017) developed a deep reinforcement learning technique that employs several asynchronous agents to simultaneously learn a policy and a value function.

Some of the drawbacks of existing reinforcement learning algorithms, such as the large variation of policy gradients and the sluggish convergence of value-based techniques, are addressed by A3C.

The A3C algorithm works by simultaneously training multiple actor-specific agents, each with its own copy of the environment and parameters. Agents asynchronously update their own weights on the network based on their local knowledge, enabling faster and more efficient training. The Actor-Critic architecture has two parts: the actor choosing the action according to the current law; and Critic, which estimates the operating cost for each state. The

A3C algorithm uses a variant of the actor-critic approach called Advantage Actor-Critic (A2C), which incorporates advantage estimation into the gradient adjustment rule. Optimization measures the benefits of taking action in a state compared to that state's needs under current law.

The A3C algorithm modifies the actor and critic network based on the following drop functions [26]:

Actor loss function:

$$L_{actor}(\theta) = -E[log\ \pi(a_t|s_t, \theta) * A_t] \quad (12)$$

Here θ is the correct parameter set, π is the correct function, a_t is the action at time t, s_t is the state at time t, and A_t is the advantage of the action at time t.

Critic loss function:

$$L_{critic}(\theta) = (V(s_t, \theta) - R_t)^2 \quad (13)$$

where V is the estimated value function of the state, R_t is the discounted sum of rewards from time t to the end of the episode, and θ is the set of critic network parameters.

The A3C algorithm then updates the network parameters using stochastic gradient descent, where each agent contributes its own gradient to the optimization process.

A3C has been shown to be effective in a variety of reinforcement learning tasks, including playing Atari games, navigating complex mazes, and controlling robots. Its parallel and asynchronous nature allows for efficient use of computing resources and faster training compared to other algorithms.

Drone navigation tasks, such as obstacle avoidance, target tracking, mapping, and surveillance, have been successfully addressed using various algorithms. The selection of a suitable algorithm depends on the constraints and requirements of the specific task at hand, with each algorithm offering its own set of advantages and limitations.

3.DRL-Based Navigation

3.1.Framework

Drone navigation is mainly focused in the three dimensions. This process involves in searching for a shortest path or even shorter than the shortest path while navigating from source to destination by avoiding the obstacles. Most of the equipments used to perform this task are very expensive, hence most of the researchers who have involved in navigation research have performed it in a 2D space.

To categorize the primary functions of drone navigation, they can be broadly divided into two categories: point-to-point (P2P) mobility and obstacle avoidance.

Moving the drone from its starting position to a predefined target place is the challenge of point-to-point mobility. Target perspective photos can be used to acquire the target position indirectly or directly using GPS or other localization methods.

Obstacle Avoidance: Obstacles can be static, dynamic, or structurally continuous. Static obstacles are those that are fixed and do not move, such as walls, trees, or buildings. Dynamic obstacles are those that can move, such as other drones, birds, or vehicles. Structurally continuous obstacles refer a bridge or corridor that is a natural part of the landscape[27,28].

The drone needs to avoid obstacles while navigating towards the target point. This can be achieved through the use of sensors, such as laser range finders, ultrasonic sensors, cameras, or other sensors.

By interacting with the surroundings, a DRL-based navigation system seeks to decide the best course of action for directing the drone to its trage position. This goal is accomplished by modelling the navigation process as Markov Decision Process(MDP), with sensor data serving as the state. Maximising the expected benefit of activities conducted with this MDP framework is the objective.

The DRL representative goes forward while avoiding problems with the role of the area and the building block map, as well as the local plan of the traditional navigation system. Deep neural networks are trained in simulated environments and then used on real drones to make real-time decisions.

In a complex environment with many persistent configuration issues, operators can fall for local vehicles. In these cases, the normal navigation system must provide additional international information to the DRL. For DRL-based navigation, the global planning module generates a sequence of intermediate way-points that serve as destination stations. This enables the integrated navigation system to effectively navigate through complex environments over extended periods of time.

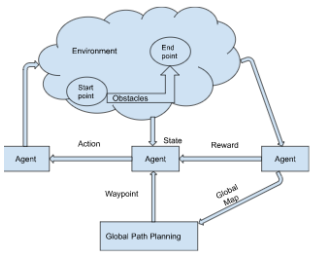

Figure 3. DRL-based Navigation.

In order to identify the optimal course of action to direct the UAV to its destination, deep reinforcement learning(DRL) algorithms are utilised for autonomous navigation of UAVs interacting with the environment. Many well-known DRL algorithms, including DQN,DDPG and PPO, as well as the algorithms they produce,have been create in order to create a DRL-based navigation system[11].

The benefits of DRL-based navigation include being mapless, having a high capacity for learning, and requiring less reliance on sensor accuracy. The physical training procedure inherently results in drone accidents with environmental barriers since RL is a technique for learning by trial and error, which is not permitted. In order to train the deep neural network for real-time navigation decision making, a simulation environment is used first.

The Deep Reinforcement Learning (DRL) algorithm is used in autonomous drone navigation to choose the optimal course of action for guiding the drone to its destination point by interacting with the surroundings. A DRL-based navigation system has been created by enhancing a number of popular algorithms arising from DRL, such as DQN,DDPG, and PPO. Some strategies model the navigation process as a Markov Decision Process(MDP), using senor data as the state, in order to accomplish the goal of maximising the expected reward of the action[11].

3.2.KeyElements

The key elements of DRL-based navigation include:

State: The state of the drone navigation system represents the current observation or perception of the environment, including information about the drone's position, orientation, and velocity, as well as information about the surrounding obstacles and other relevant features.

Action: The action refers to the maneuver that the drone performs in response to the current state. The drone can choose from a set of available actions, which are defined based on the drone's capabilities, such as moving forward, backward, up, down, or rotating in different directions.

Reward: The reward is the signal the drone receives as feedback from its surroundings after performing an action. The incentive feature is intended to motivate the drone to execute behaviours that result in the desired outcome, such travelling to a specified area while avoiding hazards. The outcome of the action can result in a positive, negative, or zero reward.

Policy: The policy is the mapping between the current state and the action that the drone should take in response to that state. A DRL algorithm is used to learn the policy via trial and error and change it in response to rewards from the environment.

Training environment: To train the drone using DRL, a simulation or real-world environment is used as the training environment. The training environment should accurately mimic the real-world conditions, taking into consideration the presence of obstacles, the impact of environmental factors such as wind, and the limitations of the drone's hardware components and sensors.

DRL algorithm: The DRL algorithm is the core of the DRL-based navigation system. The algorithm is responsible for learning a good rule of thumb that shows the current situation for

the action that leads to the best result. Popular DRL algorithms include Deep Q-Network (DQN), Optimal Policy (PPO), and Deep Deterministic Policy Gradient (DDPG).

Deployment environment: The deployment environment is the real-world environment in which the drone operates after the DRL training is complete. The deployment environment may differ from the training environment in several ways, such as the presence of new obstacles, changes in lighting or weather conditions, and variations in the drone's hardware or software. Therefore, the DRL-based navigation system should be robust and adaptive enough to operate effectively in different deployment environments.

4.Application Scenarios

Despite several years of research on DRL-based navigation, there is still no universally accepted taxonomy for such systems. One point of contention is the use of commercial localization technologies like GPS and WiFi to estimate the relative position of destination points without relying on a global map, which some researchers refer to as "mapless" navigation.

The DRL methodology, which has been referred to as a "map-based" method in prior research, involves preprocessing the drone's local observation data from its sensors into a local map, eliminating the need for global map data.

Studies have shown that while researchers may use similar DRL algorithms to solve a common problem, such as drone navigation, some have also applied domain-specific expertise and conducted specialized research for various application scenarios (as depicted in Figure 4)[28].

Figure 4. Two application scenarios of DRL-based Navigation.

These various strategies have developed because, according to the state of the art, it is challenging to converge when the DRL navigation policy space is defined too broadly. At the moment, agents typically acquire navigation skills in a given environment, which are subsequently generalised to comparable settings, to lessen the complexity of DRL training.

The DRL navigation application scenarios are divided into two groups for the purposes of this evaluation: Indoor navigation and local obstacle avoidance. Table 1 and Table 2 provide a brief comparison of different circumstances. The fundamental navigational duties are the same for each scenario, but the specifics and emphasis are different. To focus on the dynamic changes of the environment, local obstacle-avoidance scenarios are emphasized, while indoor navigation highlights the complexity of the interior structural environment.

4.1.Local Obstacle Avoidance using DRL-Navigation

4.1.1.Understanding:

Local obstacle avoidance is an important feature in drone navigation that allows the drone to navigate safely and autonomously in cluttered environments. Local obstacle avoidance refers to the drone's ability to detect and avoid obstacles in its immediate surroundings, without relying on global path planning or prior knowledge of the environment.

Some key features of local obstacle avoidance include:

Sensor fusion: Sensor fusion is often employed to combine the information from multiple sensors and create a more reliable and accurate perception of the environment. With obstacle avoidance, the drone can adjust its trajectory in real-time to avoid collisions with detected obstacles while continuing towards its destination. By fusing information from multiple sensors, the drone can overcome the limitations of individual sensors and obtain a more accurate and reliable perception of its surroundings.

Reactive control: Once an obstacle is detected, the drone must react quickly to avoid it.Reactive control refers to the capability of the drone to generate collision-free trajectories based on the current sensor information in real-time. This means that the drone can respond to changes in the environment quickly and adjust its trajectory accordingly to avoid obstacles.

Safety guarantees: Local obstacle avoidance should provide safety guarantees, ensuring that the drone can avoid collisions with obstacles while still being able to achieve its intended mission objectives.

Learning-based approaches: Deep reinforcement learning (DRL) is a popular technique used for local obstacle avoidance in drones. DRL algorithms can learn from experience and adapt to different environments, making them suitable for real-world applications.

Simple comparison of several DRL-based navigation scenarios is shown in Table 1.

Aspect	Local obstacle avoidance	Indoor navigation DRL-based
Static obstacle	Yes	Yes
Dynamic obstacle	Yes	Yes
Structured continuous obstacle	No	Yes
Obstacle scale	Small	Large
Obstacle velocity	Low	High
Cooperation	No	Yes
Randomness	No	Yes

Simple comparison of several DRL-based navigation scenarios is shown in Table 2.

Aspect	Local obstacle avoidance	Indoor navigation DRL-based
Obstacle detection	Sensor-based	Sensor-based and vision-based
Obstacle avoidance method	Reactive-based	Reactive-based and planning-based
Learning capability	Limited	High
Adaptability to environment changes	Limited	High
Navigation accuracy	Moderate	High
Complexity of system design	Low	High
Deployment flexibility	Limited	High

4.1.2.Development

To develop a navigation system for a robot, Guangda Chen et.al. (2021) proposed a methodology that involves creating a map-based environment where obstacles are represented as polygons. They extracted information from the map using a convolutional neural network(CNN), and then they trained the CNN to discover the best course of action for the robot to take in order to navigate to the goal while avoiding obstacles using the Deep Q-Network(DQN) algorithm[29,30].

The "locality-aware embedding" method, which the authors introduce, enhances the robot's capacity to navigate in challenging situations by including the robot's present state and orientation into the CNN.

The proposed approach is evaluated through simulations in various map-based environments with different obstacles, and the results show that it outperforms existing methods for map-based obstacle avoidance in terms of success rate and efficiency.

The proposed approach involves two primary components: an obstacle detection module based on map representation and a navigation module based on DRL. The map-based obstacle detection module uses sensor inputs, such as a laser scanner or stereo camera, to generate a 2D occupancy grid map that represents the environment. From the generated map, obstacle features such as the obstacle's distance, angle, and size, as well as the robot's position and velocity, are extracted.

To navigate the drone, the DRL-based navigation module takes the obstacle features and robot state extracted by the map-based obstacle detection module as inputs and outputs the best action for the drone to take. These actions could include moving forward, turning left or right, or stopping. The DRL algorithm used in this study is DQN, which learns the optimal action to take based on the current state of the drone.

To train the DRL model, the authors used a simulation environment that generates realistic indoor and outdoor environments with various obstacles. The simulation environment also includes realistic sensor models, such as laser scanners and cameras, to provide input data to the obstacle detection module. The authors demonstrated the usefulness of the suggested strategy for navigating towards the intended destination while avoiding obstacles by evaluating its performance using both simulation and real-world trials.

The authors employed a deep reinforcement learning(DRL) algorithm to generate an ideal policy for the robot to follow in their map-based approach to obstacle avoidance in mobile robot navigation. The system processed the map data and produced the robot's behaviours using a convolutional neural network(CNN).

The robot in a research used a laser range finder and a camera that were installed on it to acquire map data. The robot's position and the location of any impediments in the area may both be determined using these sensors. The authors used a local costmap to represent the obstacles and the robot's current position and orientation, which was updated in real-time based on the sensor data.

The DRL algorithm was trained using a simulation environment, where the robot interacted with the environment and learned to avoid obstacles and reach the goal location. The authors used the Proximal Policy Optimization (PPO) algorithm, which is a popular DRL algorithm for continuous action spaces.

The trained model was then tested on a real mobile robot in a simulated environment and a real-world environment. The robot successfully avoided obstacles and reached the goal location in both environments, demonstrating the effectiveness of the proposed approach.

4.2.Point-to-Point Indoor Navigation.

4.2.1.Feature

The movement of a mobile robot from an origin point to a destination point within an interior environment is known as indoor P2P navigation. The robot must be able to locate itself in space, determine its orientation, and create a path to the goal point that avoids hitting any obstacles. The following are some of the essential components of indoor P2P navigation: Localization, Path Planning, Obstacle avoidance, Robustness, and Efficiency.

4.2.2.Development

In the year (2020), Enrico Sutera et.al. have proposed a method, where the authors use an Ultra-wideband (UWB) sensor to provide precise localization information of the robot in the environment[31]. The UWB sensor is used to estimate the distance between the robot and a set of UWB anchor nodes placed in the environment, allowing for accurate localization of the robot.

Additionally, the authors used a Deep Q-Network (DQN) algorithm to figure out the best course for the robot to take in order to avoid obstacles and reach to the goal point. The output of the DQN algorithm selects the optimum course of action depending on the current state of the robot, which is represented by a raw depth picture taken by a monocular camera installed on the robot.

To train the DQN algorithm, the authors use a simulator to generate synthetic environments with varying levels of complexity and randomness. The DQN algorithm is trained using a curriculum learning approach, where the complexity of the environment is gradually increased over time as the algorithm improves.

The proposed method is evaluated on a real-world indoor navigation task that involves navigating a robot to a target point

while avoiding obstacles. The authors compare their approach with other state-of-the-art navigation algorithms and demonstrate superior performance in terms of both navigation accuracy and obstacle avoidance. The robot's state is represented as a raw depth image from a monocular camera, and a DQN algorithm is used to learn an optimal policy for navigation.

Table 3 and Table 4 depicts the simple Simple comparison of relevant references.

5.Challenges and Solution

5.1.Challenges

P2P and security protection, as mentioned earlier, are included in the operation of mobile robots. When the agent reaches the goal, he gets a good reward. This reward is rare because it is only created once at the end of each period. The broker must decide on the correction process to achieve the goal of creating a rare gift, because as we know, the value of RL depends on the sum of the value effect effect. A random search has a low chance of identifying a rare reward.

As the complexity and dynamics of the environment increase, the size of the state space also grows rapidly. Data invalidity is exacerbated by infrequent rewards, which also increase training time and poor training convergence.

5.2.Solution

The sparse reward problem is a common challenge in reinforcement learning, where the agent receives very little or no reward for most of the state-action pairs, making it difficult to learn a good policy. To address this problem, several techniques have been proposed:

Shaping the reward: This involves designing a reward function that provides more informative feedback to the agent. For example, instead of just rewarding the agent for reaching the goal, intermediate rewards can be given for making progress towards the goal or for avoiding obstacles. This can help the agent learn faster by providing more frequent feedback.

Curriculum learning: This refers to a technique where the difficulty of the task is gradually increased as the agent learns and improves its performance. This can help the agent learn faster by starting with simpler tasks and gradually moving towards more complex ones, where the rewards are more sparse [31].

Exploration strategies: This involves encouraging the agent to explore the environment more thoroughly to find more rewarding actions. This can be done using techniques such as epsilon-greedy exploration or adding noise to the action selection process.

Hierarchical reinforcement learning: This involves decomposing the task into subtasks, each with their own reward function. This can help reduce the sparsity of the rewards by providing more frequent feedback at the subtask level.

Table 3. Simple Comparison of References

Application scenario	Reference	Algorithm	Action space	Reward setting	Year
Indoor environments	[32]	DDPG	Continuous	Collision and distance-based	2021
Dynamic obstacle avoidance in cluttered environments	[33]	DRL	Continuous	Distance-based	2021
Urban navigation	[34]	DDPG	Continuous	Distance-based	2018
Indoor environments	[35].	DDQN	Discrete	Time-based	2017
Dynamic obstacle avoidance in urban environments	[36]	DRL	Continuous	Distance-based	2018

Intrinsic motivation: This involves providing the agent with an internal motivation to explore and learn, even in the absence of external rewards. This can be done using techniques such as curiosity-driven exploration or unsupervised learning.

6.Results:

In the literature review, different DRL-based approaches for autonomous drone navigation were identified. Convolution neural networks (CNNs) were utilized in some studies to extract features from sensor data, while deep Q-networks (DQNs) were utilized in others to learn navigation policies. Recurrent neural networks (RNNs) and multi-agent reinforcement learning (MARL) frameworks were also explored for drone navigation in some studies. Moreover, some studies have focused on DRL-based approaches for specific tasks such as collision avoidance, target tracking, and navigation in GPS-denied environments.

The review also identified several challenges associated with DRL-based approaches for drone navigation. For instance, the high-dimensional input space of sensor data can make it difficult to train DRL algorithms effectively. Additionally, the limited battery life of drones can constrain the amount of data that can be collected, making it challenging to train DRL algorithms from scratch.

7.Discussion:

The literature suggests that DRL-based approaches have the potential to significantly improve the performance of autonomous drone navigation. However, several challenges remain that need to be addressed. One possible solution is to use transfer learning, which enables the drone to transfer knowledge learned in one environment to another. Using transfer learning can be beneficial as it reduces the data requirement to train the DRL model and increases its performance. Additionally, data augmentation techniques can be employed to generate more training data, which can improve the accuracy of the DRL algorithm.

8.Conclusion:

In summary, the systematic review presents a comprehensive overview of the current state-of-the-art in DRL-based approaches for autonomous drone navigation, including algorithms, architectures, and applications. The review also highlights the challenges of DRL-based methods and offers potential solutions to address these issues. Additionally, the review discusses potential future directions for research in this field, such as developing more efficient DRL algorithms for real-world applications.

References:

[1] M. Y. Arafat, M. M. Alam, and S. Moh, "Vision-Based Navigation Techniques for Unmanned Aerial Vehicles: Review and Challenges," Drones, vol. 7, no. 2, p. 89, Jan. 2023, doi: 10.3390/drones7020089.

[2] R. Mur-Artal, J. M. M. Montiel and J. D. Tardós, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," in IEEE Transactions on Robotics, vol. 31, no. 5, pp. 1147-1163, Oct. 2015, doi: 10.1109/TRO.2015.2463671.

[3] W.A.S Norzam et al 2019 IOP Conf. Ser.: Mater. Sci. Eng. 705 012037 DOI 10.1088/1757-899X/705/1/012037

[4] S. Anandharaman, M. Sudhakaran and R. Seyezhai, "A low-cost visual navigation and mapping system for Unmanned Aerial Vehicle using LSD-SLAM algorithm," 2016 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, India, 2016, pp. 1-6, doi: 10.1109/GET.2016.7916802.

[5] S. Kohlbrecher, O. von Stryk, J. Meyer and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, Kyoto, Japan, 2011, pp. 155-160, doi: 10.1109/SSRR.2011.6106777.

[6] Hodge, V.J., Hawkins, R. & Alexander, R. Deep reinforcement learning for drone navigation using sensor data. Neural Comput & Applic 33, 2015–2033 (2021). https://doi.org/10.1007/s00521-020-05097-x

[7] Heidari, Hossein & Mardani, Mohammad. (2016). Applying Artificial Bee Colony Algorithm for feature optimization in UAV Navigation. INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY. 15. 6923-6932. 10.24297/ijct.v15i7.1535.

[8] M. Elbanhawi and M. Simic, Sampling-based robot motion planning: A review, IEEE Access, vol. 2, pp. 56–77, 2014.

[9] Y. Xue and W. Chen, "A UAV Navigation Approach Based on Deep Reinforcement Learning in Large Cluttered 3D Environments," in IEEE Transactions on Vehicular Technology, 2022, doi: 10.1109/TVT.2022.3218855.

[10] Amer, K., Samy, M., Shaker, M. and ElHelw, M., 2021, January. Deep convolutional neural network based autonomous drone navigation. In Thirteenth International Conference on Machine Vision (Vol. 11605, pp. 16-24). SPIE.

[11] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," in Tsinghua Science and Technology, vol. 26, no. 5, pp. 674-691, Oct. 2021, doi: 10.26599/TST.2021.9010012.

[12] J. Lin, X. Yang, P. Zheng and H. Cheng, "End-to-end Decentralized Multi-robot Navigation in Unknown Complex Environments via Deep Reinforcement Learning," 2019 IEEE International Conference on Mechatronics and Automation (ICMA), Tianjin, China, 2019, pp. 2493-2500, doi: 10.1109/ICMA.2019.8816208.

[13] C. H. Liu, Z. Chen, J. Tang, J. Xu and C. Piao, "Energy-Efficient UAV Control for Effective and Fair Communication Coverage: A Deep Reinforcement Learning Approach," in IEEE Journal on Selected Areas in Communications, vol. 36, no. 9, pp. 2059-2070, Sept. 2018, doi: 10.1109/JSAC.2018.2864373.

[14] Y. F. Chen, M. Liu, M. Everett, and J. P. How, Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning, arXiv preprint arXiv:1609.07845, 2016.

[15] Pi Y., Nath N. and Behzadan A. (2020). "Deep neural networks for drone view localization and mapping in GPS-denied environments" In: Proc. 37th CIB W78 Information Technology for Construction Conference

(CIB W78), São Paulo, Brazil, pp. 1-16. DOI:http://dx.doi.org/10.46421/2706-6568.37.2020.paper001

[16] AlMahamid, F. and Grolinger, K., 2022. Autonomous Unmanned Aerial Vehicle Navigation using Reinforcement Learning: A Systematic Review. Engineering Applications of Artificial Intelligence, 115, p.105321.

[17] Gonzalez, L.F., Montes, G.A., Puig, E., Johnson, S., Mengersen, K. and Gaston, K.J., 2016. Unmanned aerial vehicles (UAVs) and artificial intelligence revolutionizing wildlife monitoring and conservation. *Sensors*, *16*(1), p.97.

[18] Aouf, A., Boussaid, L. and Sakly, A., 2019. Same fuzzy logic controller for two-wheeled mobile robot navigation in strange environments. Journal of Robotics, 2019.

[19] C. J. C. H. Watkins, Learning from delayed rewards, PhD dissertation, University of Cambridge, Cambridge, England, 1989.

[20] Robert N. Boute, Joren Gijsbrechts, Willem van Jaarsveld, Nathalie Vanvuchelen, Deep reinforcement learning for inventory control: A roadmap,European Journal of Operational Research, Volume 298, Issue 2, 2022, Pages 401-412, ISSN 0377-2217, https://doi.org/10.1016/j.ejor.2021.07.016.

[21] M.Kannan, C.Priya, 2022, An Early detection of NIPAH Infectious Disease based on Integrated Medical Features For Human using Ensemble RBM Techniques, Journal of Pharmaceutical Negative Results (Vol. 13, No. 9)

[22] Van Hasselt, H., Guez, A. and Silver, D., 2016, March. Deep reinforcement learning with double q-learning. In Proceedings of the AAAI conference on artificial intelligence (Vol. 30, No. 1).

[23] Muñoz, G., Barrado, C., Çetin, E. and Salami, E., 2019. Deep reinforcement learning for drone delivery. Drones, 3(3), p.72.

[24] Van Hasselt, H., Guez, A. and Silver, D., 2016, March. Deep reinforcement learning with double q-learning. In Proceedings of the AAAI conference on artificial intelligence (Vol. 30, No. 1).

[25] Cetin, E., Barrado, C., Munoz, G., Macias, M. and Pastor, E., 2019, September. Drone navigation and avoidance of obstacles through deep reinforcement learning. In 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC) (pp. 1-7). IEEE.

[26] Azar, A.T., Koubaa, A., Ali Mohamed, N., Ibrahim, H.A., Ibrahim, Z.F., Kazim, M., Ammar, A., Benjdira, B., Khamis, A.M., Hameed, I.A. and Casalino, G., 2021. Drone deep reinforcement learning: A review. Electronics, 10(9), p.999.

[27] Sewak, Mohit. (2019). Actor-Critic Models and the A3C: The Asynchronous Advantage Actor-Critic Model. 10.1007/978-981-13-8285-7_11.

[28] Batra, S., Huang, Z., Petrenko, A., Kumar, T., Molchanov, A. and Sukhatme, G.S., 2022, January. Decentralized control of quadrotor swarms with end-to-end deep reinforcement learning. In Conference on Robot Learning (pp. 576-586). PMLR.

[29] Ross, Stephane & Melik-Barkhudarov, Narek & Shankar, Kumar & Wendel, Andreas & Dey, Debadeepta & Bagnell, J. & Hebert, Martial. (2012). Learning Monocular Reactive UAV Control in Cluttered Natural Environments. Proceedings - IEEE International Conference on Robotics and Automation. 10.1109/ICRA.2013.6630809.

[30] Missura, Marcell & Lee, Daniel & Bennewitz, Maren. (2018). Minimal Construct: Efficient Shortest Path Finding for Mobile Robots in Polygonal Maps. 7918-7923. 10.1109/IROS.2018.8594124.

[31] Chen, Guangda & Pan, Lifan & Chen, Yu'an & Xu, Pei & Wang, Zhiqiang & Wu, Peichen & Ji, Jianmin & Chen, Xiaoping. (2020). Robot Navigation with Map-Based Deep Reinforcement Learning.

[32] Sheela.K, C.Priya 2022, A Research on the perspective of Exploring restricted decentralized Blockchain By Applying POFE: Proof of Familiarity and Existence to reinforce multiple domains, Journal of Pharmaceutical Negative Results (Vol. 13, No. 9)

[33] Wu, Y. and Tian, Y., 2017, April. Training agent for first-person shooter game with actor-critic curriculum learning. In International Conference on Learning Representations.

[34] Rubí, Bartomeu & Morcego, Bernardo & Pérez, Ramon. (2021). Quadrotor Path Following and Reactive Obstacle Avoidance with Deep Reinforcement Learning. Journal of Intelligent & Robotic Systems. 103. 10.1007/s10846-021-01491-2.

[35] Deshpande, Aditya & Minai, Ali & Kumar, Manish. (2021). Robust Deep Reinforcement Learning for Quadcopter Control.

[36] Wu, Tung-Cheng & Tseng, Shau-Yin & Lai, Chin-Feng & Ho, Chia-Yu & Lai, Ying-Hsun. (2018). Navigating Assistance System for Quadcopter with Deep Reinforcement Learning. 16-19. 10.1109/IC3.2018.00013.

[37] Shin, Sang-Yun & Kang, Yong-Won & Kim, Yong-Guk. (2019). Obstacle Avoidance Drone by Deep Reinforcement Learning and Its Racing with Human Pilot. Applied Sciences. 9. 5571. 10.3390/app9245571.

[38] Nirmala Gururaj ,Viji Vinod . 2022 ,Deep grading of mangoes using Convolutional Neural Network and Computer Vision , Multimedia Tools and Applications ,DOI:10.1007/s11042-021-11616-2.

[39] N. RajKumar* and Viji Vinod,, 2015, Integrated Educational Information Systems for Disabled Schools via a Service Bus using SOA, Indian Journal of Science and Technology,Volume: 8, Issue: 13, Pages: 1-7, DOI: 10.17485/ijst/2015/v8i13/55030

[40] M. Sewak, S. K. Sahay and H. Rathore, "Value-Approximation based Deep Reinforcement Learning Techniques: An Overview," 2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, 2020, pp. 379-384, doi: 10.1109/ICCCA49541.2020.9250787.

[41] Z. Jin, J. Wu, A. Liu, W. -A. Zhang and L. Yu, "Policy-Based Deep Reinforcement Learning for Visual Servoing Control of Mobile Robots With Visibility Constraints," in IEEE Transactions on Industrial Electronics, vol. 69, no. 2, pp. 1898-1908, Feb. 2022, doi: 10.1109/TIE.2021.3057005.