Journal of Theoretical and Applied Information Technology
15st June 2026. Vol.104. No.11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

CAN REINFORCEMENT LEARNING ENHANCE 3D AERIAL ROBOTIC EXPLORATION AND NAVIGATION IN COMPLEX OBSTACLE-RICH ENVIRONMENTS? AN OPTIMISED SARSA-BASED LEARNING APPROACH

VINOD K S[1]*, E D KANMANI RUBY[2]

[1] Research Scholar, Vel Tech Rangarajan Dr. Sagunthala R & D Institute of Science and Technology, Chennai, 600062, Department of Electronics and Communication Engineering, India

[2] Professor, Vel Tech Rangarajan Dr. Sagunthala R & D Institute of Science and Technology, Chennai, 600062, Department of Electronics and Communication Engineering, India.

E-mail: [1]vinodksece@gmail.com, [2]bewinbewin54@gmail.com

*Corresponding Author: Vinod K S

ABSTRACT

Aerial Robotic exploration and navigation play an important role in finding a safe and optimum route from goal position to destination point. In this paper, an Optimized State Action Reward State Action (SARSA) reinforcement learning algorithm is proposed for autonomous navigation of UAVs in complex 3D obstacle environments. A comparative study is carried out between existing traditional SARSA algorithm and proposed optimized SARSA on real-time aerial robotic application use traditional SARSA algorithm. The traditional algorithm exhibits limitations in its performance in terms of convergence rate, exploration-exploitation balance and computational complexity, whereas the proposed algorithm includes parameters such as adaptive learning rate, decaying epsilon-greedy policy and reward shaping which leads to improvement in learning efficiency and navigation performance. To evaluate autonomous drone navigation, a 3D grid-based simulation with spatial boundaries and dynamic obstacles has been developed. The environment is designed to test flight performance in indoor spaces, search and rescue situations and rough terrains. In this study, a traditional SARSA agent with fixed hyper parameters is compared with a proposed Optimised SARSA agent with adaptive learning mechanisms. The results shows that the embedded optimisation technique greatly improve the reinforcement learning performance in aerial robotics with 55% improvement in efficiency, 52 times faster converge and 70% increase in stability. Future work will investigate model-based approaches and deep reinforcement learning to address real-world operational challenges.

Keywords: *Reinforcement Learning, SARSA, UAV Navigation, 3D Grid Simulation, Hyperparameter Optimization*

1. INTRODUCTION

Unmanned Aerial Vehicles (UAVs), commonly known as drones, are versatile platforms used across a wide range of industries. UAVs provide critical support for military operations, emergency response, precision agriculture, and aerial imaging particularly in the safe exploration of hazardous or inaccessible locations. The main research focus on autonomous navigation in complex, dynamic environments such as 3D urban landscapes, indoor search missions and structural disaster analysis. UAV technology has the ability to autonomously navigate constricted, crowded spaces where human intervention is practically not possible. Autonomous navigation for UAVs has lot of challenges in the real-world scenario, which is dynamic and constantly shifting, witnessing the presence of dynamic obstacles, diverse terrain types, and areas with inefficient GPS coverage, inherently demanding the need for more effective decision-making mechanisms. Currently, it is observed that conventional rule-based navigation algorithms in dynamic scenarios, exhibits inefficient path finding paving way for the research and

development incorporating AI approaches, particularly in reinforcement learning which is considered to be the highly promising tools. The agent uses the reinforcement learning approach which learns from mistakes to encounter and improve the performance over time, interacting with its environment and receiving rewards and punishments for its actions.

Among various RL algorithms, SARSA is relatively simple and computationally efficient. Unlike off-policy algorithms such as Q-learning, SARSA operates on-policy, which means it learns about the policy, including the exploration steps taken by that policy This capability translates into much more stable and safer trajectories, which is essential in a real-time deployment of UAVs in safety-critical tasks. SARSA differs from other algorithms, such as Q-learning, in that it updates its value function based on the action actually taken by the current policy. This lets it be more conservative in uncertain environments, which is a key advantage for tasks involving physical systems like UAVs where too risky an exploration can lead to damage or failure.

In this context, the present work studies the performance of some SARSA-based algorithms for 3D navigation on UAVs. The paper studies the effect of learning parameter tuning and exploration strategy modification on the learning efficiency and navigation capability of UAVs. Testing the hypothesis that a carefully tuned SARSA agent, with metric tuning such as learning rate, discount factor and epsilon decay schedule can significantly outperform a baseline, unoptimized agent in terms of convergence rate, path optimality and robustness in obstacle rich environments.

The Unmanned Aerial Vehicle (UAV) technology has gone beyond the military use and is now an essential part of the industrial and academic research. Autonomous flight control allows for real-time data processing and versatile payload capacities across many areas, including precision agriculture, infrastructure inspections, emergency response, and logistics. The performance of the UAV agents is simulated through 3D animation and performance graphs to visualize the scenario. By visualizing the paths taken by the UAV in an episode, an ability is given to observe the progress of the learning algorithm from random movement to optimized movement to reach the objective.

This work makes four contributions: (i) High-Fidelity Testbed , an interactive 3D environment for reinforcement learning-based UAV exploration, a platform for future research and development (ii) Comparative Hyperparameter Analysis to provide a comprehensive comparison between traditional and optimized SARSA implementation to describe the policy convergence (iii) Validation of Conservative Exploration to exhibit the practical advantages of deploying SARSA in safety-critical, resource-constrained aerial applications where exploration is the main focus(iv) Extensible Framework to give a robust foundation for autonomous aerial navigation systems through reinforcement learning.

This research paper is structured as follows: Section 2 is the related or existing work in the area of UAV navigation, various reinforcement learning algorithms, in particular, the SARSA algorithm. Section 3 explains the traditional approach, Sections 4 highlights an explanation of proposed optimized SARSA including the implementation of simulated environment, algorithm, and the methods of optimization and 5 discuss the virtual implementation of proposed approach to the research, including the software and agent architecture. Section 6 is the result analysis discussion and section 7 gives the limitation of SARSA, section 8 discusses difference between prior work and current work, section 9 gives the open research issues from finding from current work and section 10 concludes the paper and gives the future directions of research.

2. RELATED WORK

Reinforcement learning (RL) has also proven promising for the navigation of aerial robots in complex settings without using pre-programmed knowledge of the environment. Out of various popular classical reinforcement learning algorithms [1], one such algorithm named SARSA has been identified for its efficiency and real-time learning capabilities in discretized 3D grid space. Recent advances have shown the improvement of SARSA through various methods, such as parameter adjustments and combinations. For example, the improvement brought by the dynamic epsilon-greedy strategy, adapting the learning rate, and two-player extensions has been found to be efficient for dense obstacle environments. Furthermore, researchers have explored various methods of improvement, including reward shaping, transfer learning, and state abstraction for faster learning.

Drones are gaining increasing use; hence, the trend is a continuously growing area of interest. Of the many technologies, quadrotors make one of the

most versatile and broadly applicable classes of drones, with several of their applications requiring autonomy. The challenges range from maintaining stability through path planning to maintaining control during autonomous manoeuvres. Conventional control algorithms such as PID controllers often fall short on their tuning parameters. Recently, machine learning has gained significant attention due to its potential to enable autonomous navigation of UAVs to their destinations. In paper [2], the authors configured a drone for autonomous navigation by utilizing agents along with machine learning algorithms that are capable of moving the drone through a virtual physical environment. The quadrotor is required to fly from a start position to a target goal location. The agent receives a reward for succeeding in moving the drone toward the goal position, but on the contrary, it receives a penalty for failing to do so. Two reinforcement learning algorithms, namely Q-learning and SARSA, and a Deep Q-Network as an agent were used. ROS with Gazebo is utilized for the simulation environment's development to implement the learning algorithm in a real-life environment. From the results, it is observed that the DQN algorithm, the AdaDelta algorithm for optimization, is the most optimal way to navigate the drone to move from the start position to the desired location.

Deep Reinforcement Learning (DRL) can be used in multiple agent scenario to help develop successful strategies for the need to work together to complete various tasks. The work [3] addresses the challenge of controlling a swarm of UAVs using Multi-Agent DRL (MARL). This coordinated movement of the drone swarm can be managed both centrally and in a distributed manner, and both approaches are explored in this study. A dynamic military scenario where a swarm of drones is tasked with demolishing enemy targets while avoiding collision. Given that the UAVs have limited battery capacity, the research prioritizes minimizing the mission time. To achieve this, a continuous-time PPO algorithm GLIDE [3] was proposed. In GLIDE, the drones collaborate with one another and communicate with a central control to determine the best actions to take. The action control in GLIDE can be managed both centrally and decent rally, using two proposed algorithms: C-GLIDE and D-GLIDE. The UAV-SIM simulator was developed, in which mines are randomly placed at unknown 2D locations for the UAVs at the start of each episode. Both the algorithms were evaluated through simulation experiments. Both C-GLIDE and D-GLIDE converge and show comparable performance in terms of target destruction rates for the same number of targets and mines. Notably, it is found that D-GLIDE is better compared to C-GLIDE

For autonomous UAVs, connectivity-aware path planning is crucial. While Reinforcement Learning (RL) algorithms are commonly used, they often converge slowly. The paper [4] proposes a transfer learning approach, utilizing a previously trained policy from an old task to enhance path learning in a new, related task. As the agent interacts with the environment, it refines its path. It evaluates this approach using sub-6GHz and mmWave tasks by modelling the connectivity-aware path problem as a constrained MDP. A model-free DQN based on Lyapunov is utilized to design the path at sub-6 GHz, guaranteeing the satisfaction of connectivity constraints. The results have demonstrated that such a method can offer substantial training time reduction in diverse urban scenarios.

UAVs have revolutionized the world of monitoring, delivery of packages, agricultural processes, and relief activities. The autonomy of UAVs relies on a number of factors. Optimization of paths is one such factor since it aims at finding the best paths considering the objectives of the task and the use of energy. The article "UAV Path Planning Techniques" summarizes different approaches for finding paths using UAVs. Probability-based models, mathematical models, and bio-inspired models are discussed. The article highlights the application of artificial intelligence, specifically machine learning, in managing dynamic paths. Finally, it refers to ways to make it more energy-efficient to fly multiple drones. Though progress has been made in this area, factors such as high complexity and reactivity issues are yet to be overcome. This article sheds some knowledge on how to design paths for UAVs in the coming future.

The paper [6] gives an overview of the drone swarms navigation, covering traditional and innovative reinforcement learning algorithms. The authors present a hybrid AI model that combines DRL in a centralized swarm agent designed to observe ground targets in a surveillance role. Tasks are distributed to UAVs from a centralized controller and are governed by sub-agents in a cooperative framework, trained from proximal policy optimization algorithms. The model includes criteria to measure performance, which showed efficiency in tracking targets during simulations.

The research [7] describes the critical path planning techniques for a UAV, along with the frameworks and simulation environments. The research has classified RL algorithms according to the environment, characteristics, ability, and application. This classification will help researchers to identify the suitable RL algorithm for the path planning process for the UAV.

The progress in drone technology has also been affected by technological changes in the field of controls. A study [8] highlights the difficulties in drone operation and explores the application of reinforcement learning for automated functioning. The implementation of Q-learning and SARSA algorithms has helped the author compare the results in auto functions in drones. The simulated data has confirmed that Q-learning has superiority over SARSA in training it.

In non-GPS navigation, the weight of the heavy sensors poses a significant constraint in autonomous indoor navigation, even though the GPS precision in UAV navigation is limited. The paper titled [9] proposes an autonomous drone system to navigate indoors to detect targets, such as an unknown box, using a camera. The authors employed a DRL method to develop an adaptive control strategy that mimics the techniques used by expert pilots. The usefulness of their system is shown through simulation experiments on various indoor scenarios, while visualization is used to better understand the trained network. In addition, they proposed an adaptive control algorithm to simultaneously handle multiple UAVs in collaborative object retrieval in confined spaces. The paper enhances indoor multi-drone systems, particularly by improving adaptability in both general terms and within deep reinforcement learning (DRL)

The paper [10] proposes a route planning and obstacle avoidance system in UAVs based on the use of Deep-Sarsa. The Deep-Sarsa algorithm is an on-policy RL algorithm that uses feedback from environment to assist UAVs in avoiding obstacles in a dynamic environment. The proposed algorithm was evaluated using GAZEBO and ROS software and was able to provide guidance to targets without any collisions after being trained in a grid world environment. The proposed algorithm is the first to use Deep-Sarsa in route planning in UAVs to avoid obstacles in a dynamic environment.

Task offloading in edge computing (EC) is an area that serves as a backbone in optimizing resource usage and improving system performances. The research paper [11] explores different AI techniques in computation offloading (CO) techniques and presents a hybrid learning algorithm that outperforms existing reinforcement learning techniques in terms of lower latency delay in Q-Learning and SARSA. The study proposes an application of existing literature in relation to QL and SARSA techniques in the realm of CO and mobile edge computing. The proposed hybrid adaptive approach in RL explores the current states while taking each decision in a system. The proposed strategy for exploration and exploitation helps effectively diminish latency delay in edge computing.

In the paper [12], the application of drones in route planning and obstacle avoidance is analyzed for improving the efficiency of work and the safety of flights. One significant area where drones are used is in the detection of aquaculture sea cages, which are large and dispersed in the form of a net cage and require energy-efficient planning for drones to perform their operation and then return to the home point. For route planning, the researchers concentrate on the Q-learning model and compare it with the SARSA model. For obstacle avoidance, the researchers adopted a similar reinforcement learning model.

From the literature study, there are several limitations despite of the substantial advancements made in UAV autonomous navigation employing RL and DRL techniques. Classical SARSA algorithms enable stable, safe on-policy learning, but are slow to converge and do not scale well to complex 3D environments. Deep RL methods can improve navigation intelligence and adaptability, but they also come with high computation cost, training complexity and energy consumption, which are not suitable for resource-limited UAV platforms. Moreover, the existing works mainly focus on either 2D navigation, swarm coordination or simulation-specific evaluations and do not well discuss optimised SARSA-based learning for real-time 3D aerial robotic exploration in dense obstacle-rich environments.

Therefore, an optimised SARSA-based reinforcement learning framework is important to improve the convergence speed and effectively balance the exploration and exploitation. The computational complexity is reduced and is allowed for the efficient autonomous navigation in complex 3D aerial environments. Such a framework may bridge the gap between lightweight classical RL approaches and computationally intensive DRL

models, while supporting real-time UAV deployment in practical obstacle-dense scenarios like disaster response, indoor navigation and autonomous exploration.

3. TRADITIONAL SARSA

SARSA is a TD based on-policy RL algorithm in which the agent learns a policy to estimate the Q-values of the current action and current state, based on the next state and the next action that the agent would take (hence the term "on-policy"). In the context of 3D aerial robotic exploration, the robot operates within a discrete 3D grid environment (X, Y, Z) and aims to find a goal location while avoiding obstacles. The robot utilizes SARSA to learn the optimal path through repeated exploration episodes.

In 3D grid environment a state given by $s = (X, Y, Z)$ is the current position of the robot in the 3D grid. The action includes movement in any of six possible directions. The agent can move in 6 possible directions namely, (+X, -X) for right or left; (+Y, -Y) for forward or backward; and (+Z, -Z) for up or down direction. For each action the agent gets the reward based on reward function. The reward function is defined such that when the robot or agent reaches goal it gets reward (+100 Points), if it hits an obstacle, it gets reward of (-100 points) that is there is penalty of 100 points and for each step there will be reward of -1 points. The Policy is updated using the update rule which was presented in [1] and is shown below:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)] \quad (1)$$

Here, **r** is the reward earned, **α** is the learning rate, and **γ** is the discount factor. The agent updates next state and action as **s←s′, a←a′** (see [1]). The process continues until the agent reaches a terminal state which marks the end of the episode.

There is a possibility of exploration versus exploitation dilemma which makes the algorithm to get stuck in local optima and thus the condition where the agent never reaches the goal. The epsilon-greedy approach is a simple yet effective strategy in reinforcement learning for balancing exploration and exploitation. It balances choosing the action that gives the biggest estimated reward (exploitation) and choosing a random action (exploration) with a probability epsilon (ε). This balance helps the agent discover potentially better actions and preventing the problem of getting stuck in local optima. The policy is updated based on the above update rule. The policy is also updated for every episode and the process is repeated till the goal is reached.

Figure 1 gives the block diagram of the SARSA algorithm. First Q values are initialised. The algorithm runs for multiple episodes (one episode equals one complete run from start to terminal state). At the start of each episode, the initial state, **s** is chosen (usually the starting position of the agent). The agent selects an action based on the **ε**-greedy policy: With probability 1-Ɛ choose the best action (highest Q-value). With probability Ɛ choose a random action (exploration). The following steps repeat until the episode ends (i.e., until a terminal state is reached). The agent executes action ***a,*** in the environment

The environment returns a reward **(r)** for that action and the agent moves to the next state **(s').** In the new state **(s`)**, the agent again selects an action (a') following the same ε-greedy policy. The Q-value for the current state–action pair is updated using the SARSA update rule.

The SARSA algorithm for 3D Navigation and exploration of aerial robot from start position to goal position avoiding Obstacles implemented using python is shown in figure 2. The path followed by the agent is given by blue lines, the obstacles are shown by orange and goal position is given by green star. It is seen that the Agent moves through random paths initially and then learns the policy till it reaches the goal, avoiding the obstacles.

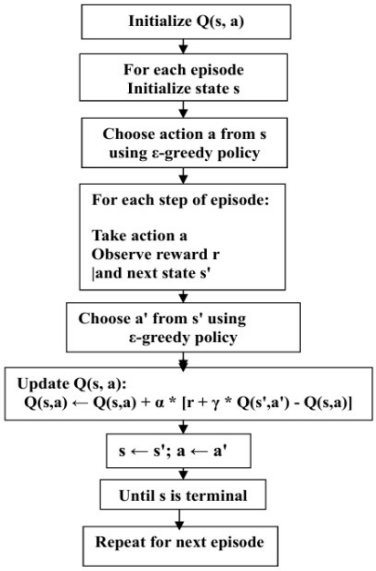

Figure 1: Flow Diagram For SARSA Algorithm

Journal of Theoretical and Applied Information Technology
15st June 2026. Vol.104. No.11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

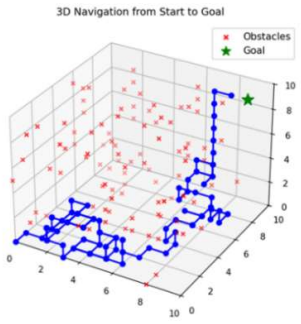

Figure 2: 3D Navigation Of Aerial Robot From Start Position To Goal Position Avoiding Obstacles Using Traditional SARSA

There is need for reducing the number of steps and also to reduce the time taken for learning. There is need for improving the overall performance for the SARSA algorithm. There is need for an optimized form of the above algorithm, so that the algorithm converges fast and performs better in all aspects. To make this possible we propose an optimized form of SARSA algorithm which is described in next section

The traditional SARSA algorithm (see [1]) can be summarized as below.

Traditional SARSA algorithm:

Initialize **Q(s, a)** arbitrarily
For each episode **do:**
  Initialize state **s**
  Repeat:
    Choose action a from state **s** using ε-greedy policy
    Take action **a**
    Observe reward **r** and the next state **s'**
    Choose action **a'** from state **s'** using ε-greedy policy
    Update **Q(s, a):**
      Q(s, a) ← Q(s, a) + α * [r + γ * Q(s', a') − Q(s, a)]
    s ← s'
    a ← a'
  Until s is a terminal state
End For

4. PROPOSED WORK

In 3-D, obstacle-rich environments like collapsed buildings due to disasters like earth quake, traditional SARSA struggles to converge quickly and learn efficiently due to sparse rewards, large state-action spaces, and inefficient exploration. Optimized SARSA introduces enhancements such as dynamic or adaptive learning rate, decaying epsilon value, reward shaping technique, early termination conditions. The Problem Setup consists of environment which is a 3D grid *(X, Y, Z)*, agent which is a UAV navigating from a start to a goal location. The state: $s = (x, y, z)$ and agent does actions which consists of 6 discrete motions: up/down/left/right/forward/backward. The objective is to reach the goal while avoiding obstacles and minimizing steps.

For optimizing SARSA the following are used

Decaying Epsilon-Greedy approach is used in which the probability (ε) is varied with time t according to the formula as in [13] and is given below:

$$\epsilon_t = \epsilon_{t0}.exp(-decay\_rate.t) \quad (2)$$

The unoptimized SARSA agent uses constant learning rate (α), discount factor (γ), and epsilon (ε) in epsilon-greedy policy. In contrast, the optimized agent employs a decaying epsilon schedule and adaptive learning rates, which allow the agent to explore more initially and gradually exploit learned knowledge. The agents update their action-value (Q) tables using the standard SARSA update rule. This balances the exploration and exploitation.

The Optimized SARSA is described below:

Optimized SARSA algorithm

Initialize **Q(s, a)** arbitrarily
Set initial **ε = ε₀**
Set initial learning rate **α0**
For each episode **do:**
  Initialize state **s**
  Set time step **t = 0**
  Repeat:
    Compute decaying exploration rate:
    εₜ = ε₀ * exp(-decay_rate * t)
    Choose action a from state **s** using ε-greedy policy with **εₜ**
    Take action **a**
    Observe reward **r** and next state **s'**
    Choose action **a'** from state **s'** using ε-greedy policy with decaying **εₜ**
      Compute adaptive learning rate:
      αₜ = α₀ / (1 + decay_factor * t)
    Update **Q(s, a):**
      Q(s, a) ← Q(s, a) + αₜ * [r + γ * Q(s', a') − Q(s, a)]
    s ← s'
    a ← a'
    t ← t + 1
  Until s is terminal
End For

Adaptive Learning Rate (α) (see [14]) is used where **l**earning rate decays with episode number

$$\alpha_t = \frac{\alpha_0}{1+decay\_factor.t} \quad (3)$$

The above formula is used for calculating the adaptive learning rate. Here α_0 is the initial or previous learning rate, α_t is the current or present learning rate.

Journal of Theoretical and Applied Information Technology
15st June 2026. Vol.104. No.11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The simulation environment consists of a 3D grid world populated with randomly placed static obstacles. The UAV agent starts from a predefined location and must navigate to a designated goal. Movement is allowed in six directions: forward, backward, left, right, upward, and downward. The environment rewards the agent for reaching the goal and penalizes for wrong action.

Reward Shaping is used to optimize the rewards. In reward shaping reward point of +100 is awarded for reaching the goal, -100 points (penalty for hitting the obstacle, -1 for each move, -10 for revisiting the same state and -5 for boundary violation for collisions or unnecessary steps. The same update rule is used as in traditional SARSA but with the redefined parameters.

Figure 3 shows the flow diagram of proposed optimized SARSA algorithm. The static exploration probability epsilon is replaced by decaying epsilon, learning rate is replaced by adaptive learning rate, the reward function is shaped as indicated before. Heuristic Guidance is used to bias exploration slightly toward goal as defined earlier.

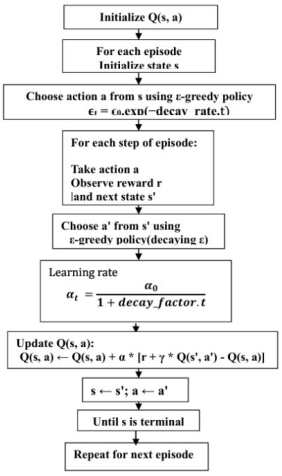

Figure 3: Flow of Optimized SARSA Algorithm

Figure 4 shows 3D Navigation of aerial robot from start position to goal position avoiding obstacles using optimized SARSA. The grid world consists of 10x10x10 3D grid. The blue line indicates the path followed by the aerial robot. The green star represents the goal position and orange crosses represents the obstacles. It is seen that that no of steps is reduced before the goal position is reached. The number of loops in the path is reduced almost to zero as compared to traditional SARSA.

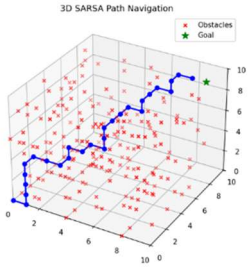

Figure 4: 3D Navigation of aerial robot from start position to goal position avoiding Obstacles using Optimized SARSA

5. VIRTUAL IMPLEMENTATION DETAILS

Both agents were developed in Python using NumPy and Matplotlib for data handling and visualization. The simulation was run over 300 episodes per agent, with each episode limited to a maximum number of steps. The enviro ment state space was represented by the current 3D grid location of the UAV, while the action space is the set of possible movements. Each step involved state transition, reward calculation, and Q-value updates. For the optimized agent, epsilon was decayed exponentially over time, and the learning rate was adjusted based on the improvement in performance metrics over episodes. This allowed the agent to reduce exploration once a satisfactory policy was emerging.

6. RESULT ANALYSIS

To study the performance 5 key metrics were used. The average reward per episode indicates how well the agent is learning over time. While the unoptimized agent reached flat level earlier, the optimized agent displayed a steadily increasing trend.

The cumulative measure of overall policy effectiveness is given by higher total reward which the optimized agent exhibits. Steps per episode, is a gauge of navigation effectiveness.

The optimized agent revealed reduced the number of steps required to accomplish the goal more quickly over time. The optimized agent consistently discovered shorter paths, and the minimum steps to goal metric records the most

effective episode. Finally, the trade-off between exploration and exploitation is demonstrated by the epsilon decay trend. While the static traditional agent continued to behave poorly, the optimized agent showed effective transition from exploration to exploitation.

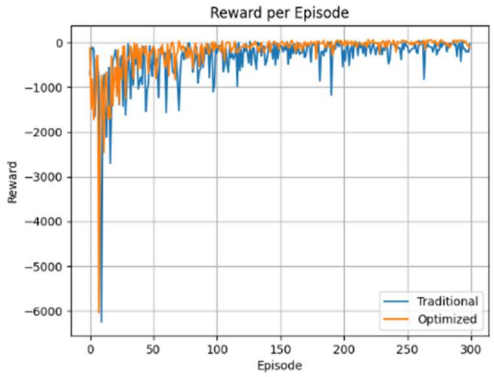

Figure 5: Reward Vs Episode

Table 1: Comparison Of Traditional And Optimized SARSA Rewards Per Episode Vs Episode

Cumulative Reward per Episodes Vs No of Episodes		
Episode	**Cumulative Reward Traditional**	**Cumulative Reward Optimized**
0	-443	-272
1	-1015	-550
2	-3229	-1304
3	-4313	-2803
…	…	…
…	…	…
…	…	…
296	-101560	-57835
297	-101709	-57775
298	-101707	-57769
299	-101915	-57815

The performance of two SARSA agents over 300 episodes is compared in the "Reward per Episode" graph in Figure 5. The optimized SARSA agent is represented by orange, while the conventional SARSA agent is represented by blue. The traditional SARSA agent starts out with noticeably negative rewards during the first performance period (from episodes 0 to 50), frequently falling below -2000 and even reaching lows below -6000. The optimized SARSA agent, on the other hand, begins with low rewards as well, but they stabilize faster. Additionally, in comparison with the standard SARSA algorithm, the optimized one has less extreme negative spikes.

During the initial episodes of learning (episodes 50 to 150), both agents made promising progress. The Optimized SARSA, though, had a quicker learning rate, with its rewards slowly but steadily progressing towards the optimal value (zero), than the Traditional SARSA. Regarding their performances in the long run (episodes 150 to 300), the optimized agent performed remarkably well with its rewards perfectly on track. The traditional one was unpredictable with moderate to inconsistent performances.

Table 1 above shows that the optimized SARSA algorithm is performing better compared to the conventional algorithm in terms of convergence. It discovered the optimal actions much faster and performed in a rather consistent manner. The rewards achieved by the algorithm in each episode were rather consistent, and this means the learning and performance were stable. The average reward for the conventional SARSA algorithm is -339.717, and this value is reduced to -192.717 for the optimized SARSA.

Optimized SARSA performs better, and this improvement is due to several reasons. First, using the adaptive learning rate helps in making the learning process stable over time. Also, the use of the decaying epsilon value helps find the best compromise between both exploration and exploitation processes.

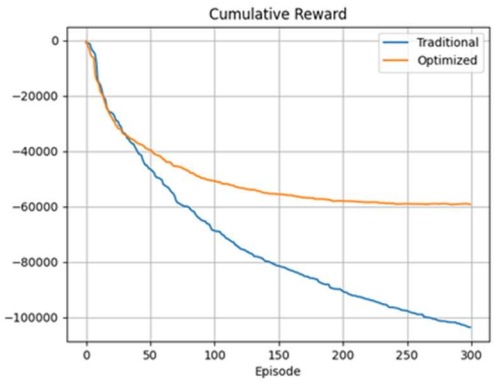

Figure 6: Cumulative Reward Vs Episode

Table 2: Cumulative Rewards per episodes vs No of episodes for Traditional and Optimized SARSA

Rewards per Episode Vs Episode		
Episode	**Reward per Episode Traditional SARSA**	**Reward per Episode Optimized SARSA**
0	-443	-272
20	-1659	-858
40	-873	-377
60	-620	-227
80	-525	-447
100	-117	44
120	-91	18
140	-51	-158
160	-126	59
180	-41	-168
200	-200	49
220	-51	59
240	-178	21
260	-130	64
280	-276	66

Additionally, using reward shaping helps in getting more information about the learning process, which assists in improving learning. Also, state visitation heuristics help explore the least visited states, while using termination conditions helps terminate unnecessary explorations.

A comparison between the traditional SARSA and optimized SARSA agents over 300 episodes based on their rewards is carried out using a graph of the cumulative reward against the number of episodes, as shown in Figure 6.

From the Cumulative Reward chart, both agents are close to the zero mark, acquiring negative rewards at the start, which denotes punishment for the wrong actions. In the traditional SARSA agent, denoted by the blue line, the acquisition of negative rewards is steeper, reaching close to -110,000 by episode 300. On the other hand, the optimized SARSA agent, denoted by the orange line, acquires fewer negative rewards, which levels off at around -60,000. This is roughly an improvement of 45% compared to the traditional SARSA agent.

The convergence behavior of traditional SARSA follows a smooth, steep decline in cumulative reward. This may point to the continuation of poor decision-making and low-speed learning or persistent high-cost exploration. Optimized SARSA holds a curve that begins to flatten after approximately 150 episodes—a sign of converged learning and a stable policy settled toward the avoidance of high penalties.

Table 2 gives Cumulative Rewards per episodes vs No of episodes for Traditional and Optimized SARSA the cumulative rewards per episodes for traditional SARSA is -101915 while for optimized SARSA it is reduced to -57815.the cumulative rewards per episode is reduced in optimized SARSA.

The Optimized SARSA agent has many advantages and benefits from several enhancements, such as an adaptive learning rate that improves convergence speed. Additionally, reward shaping encourages favorable transitions while penalizing undesirable ones. Epsilon decay effectively balances exploration and exploitation dilemma. Finally, early termination helps the agent to learn safer and more reward-efficient policies early in training, resulting in a more gradual cumulative reward decline.

The curve shown in Figure 7 depicts the number of steps per Episode vs number of episodes taken by Traditional SARSA agent (blue line) and optimized SARSA agent (orange line) over 300 episodes.

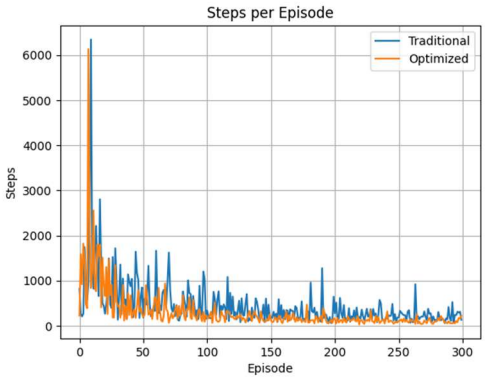

Figure 7: Steps Vs No. of Episodes

Table 3: Steps per episodes vs no of episodes for traditional and optimized SARSA

Steps per episodes vs no of episodes		
Episode	**Steps Traditional**	**Steps Optimized**
0	544	373
20	1760	959
40	974	478
60	721	328
80	626	548
100	218	57
120	192	83
140	152	259
160	227	42
180	142	269
200	301	52
220	152	42
240	279	80
260	231	37
280	377	35

In the meantime, within the initial stage (episodes 0–50), a large number of steps are taken by both agents, sometimes in excess of 6,000 in each episode, which represents inefficient navigation and exploration. But the optimized SARSA shows faster convergence in the number of steps taken compared to the traditional SARSA agent.

In the learning phase (episodes 50 to 150), it is observed that the number of steps required to learn in the Optimized SARSA algorithm is constantly less compared to Traditional SARSA. This indicates that the Optimized SARSA algorithm converges more effectively towards goal points and demonstrates improved path planning.

Both agents exhibit stabilization in episodes 150 to 300. However, the optimized SARSA model, maintains a consistently lower average step value and is more stable. Traditional SARSA, on the other hand, displays fluctuations, which can be considered points of inefficiency in exploration or decision-making. Optimized SARSA learns better policies, enabling it to accomplish tasks faster. This leads to less time spent per episode, less time with negative rewards or risk, and less computational effort.

On the other hand, traditional SARSA has slower convergence, larger variance, and a continuing process of inefficiency. These issues are likely due to its static learning rate, lack of adaptive exploration control, and absence of state visitation heuristics or early termination mechanisms.

Table 3 gives Steps per episodes vs no of episodes for traditional and optimized SARSA from

the table it is seen that the average no of steps per episodes is 440.717 for traditional SARSA while it is reduced to 293.717 for optimized SARSA for 300 episodes

The "Steps per Episode" plot further supports the effectiveness of the optimized SARSA approach. By minimizing the number of steps through adaptive strategies and early termination, the optimized agent not only reaches goals more efficiently but also does so with greater consistency, an essential quality for aerial robotics and navigation in resource-constrained environments.

Figure 8, titled "Minimum Steps Achieved," compares the best performance (in terms of minimum steps) achieved by both Traditional SARSA and Optimized SARSA agents over 300 episodes.

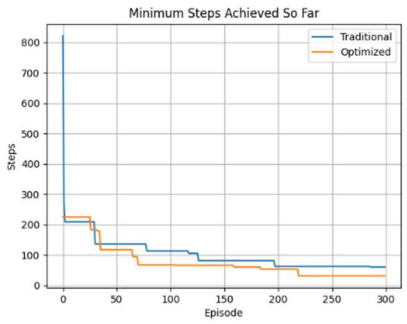

Figure 8: Min Steps Achieved Vs Episodes

Table 4: Minimum steps Achieved per episode vs no of Episode for traditional and Optimized SARSA

Minimum steps Achieved per episode vs no of Episode		
Episode	**Steps Traditional SARSA**	**Steps Optimized SARSA**
0	544	373
20	208	200
40	173	137
60	173	120
80	137	60
100	137	57
120	96	57
140	96	54
160	96	42
180	96	41
200	48	41
220	48	41
240	48	41
260	45	37
280	45	35

During the initial phase (episodes 0–50), both agents start with relatively high minimum step counts. Traditional SARSA begins with a notably high count of approximately 820, indicating a poor initial policy. Optimized SARSA begins at approximately 220 and progresses to better improvements in the number of step decreases sooner than in traditional SARSA.

Optimized SARSA is always finding new minimum values sooner than traditional SARSA. By episode 50, optimized SARSA is already finding a minimum of 70 steps, while traditional SARSA is still above 100 steps. Stepwise drops are when a new optimal policy is found.

Regarding the cumulative performance from episode 250 through 300, the optimized SARSA agent achieves a lower cumulative minimum number of steps of around 35 steps, while the traditional SARSA agent plateaus at a higher cumulative minimum of around 65 steps, thereby being unable to catch up. This implies that the optimized agent learns faster and discovers better routes to the target.

The given graph displays a monotonic decrease, signifying a consistent tracking of improvement. Optimized SARSA learns faster because it recognizes optimal routes early, performs better regarding best-case optimization with a smaller number of steps to goal attainment, and makes smoother improvement in policy with less plateau and more breakthroughs.

Table 4 shows the minimum steps achieved per episode vs. the number of episodes in traditional SARSA and optimized SARSA. In the above-mentioned table, it can be seen that the average minimum number of steps achieved per episode in traditional SARSA is 116.773, which is further reduced to 79.756 in SARSA optimization.

This gain is probably due to heuristics in exploration strategies, reward shaping to promote the use of shorter paths, and a decaying epsilon value that enables a smooth transition from exploration to exploitation.

From the graph, the plot confirms that the optimized SARSA algorithm not only converges better, as it took fewer iterations to converge, but it also resulted in more optimal policy routes in terms of the shortest possible path. All these make it an excellent candidate algorithm in terms of robotic path navigation.

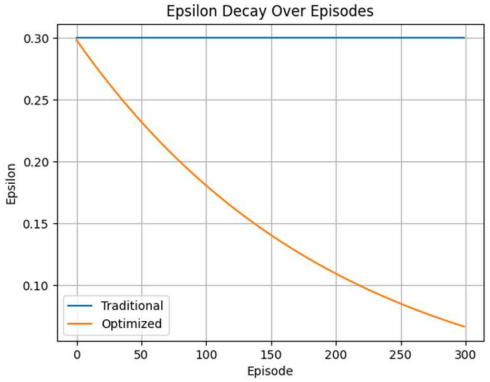

Figure 9: Epsilon Vs Episodes for traditional and optimized SARSA

Table 5: Epsilon versus No of episodes for Traditional and Optimized SARSA

Epsilon vs no of episodes		
Episode	**Epsilon Traditional**	**Epsilon Optimized**
0	0.3	0.2985
20	0.3	0.270026228
40	0.3	0.244268556
60	0.3	0.220967896
80	0.3	0.199889874
100	0.3	0.180822475
120	0.3	0.163573906
140	0.3	0.14797067
160	0.3	0.133855819
180	0.3	0.121087376
200	0.3	0.10953691
220	0.3	0.099088237
240	0.3	0.089636257
260	0.3	0.081085898
280	0.3	0.073351153

Figure 9, titled "Epsilon Decay Over Episodes," compares the evolution of the exploration rate (epsilon) in Traditional SARSA and Optimized SARSA implementations across 300 episodes.

In Traditional SARSA, with the blue line, a constant epsilon of 0.3 is used. This implies that the probability of selecting a random action is 30% at all times, irrespective of how much an agent has learned. This is not an adaptive strategy, and this might cause it to stick to stochastic optimal behavior for an extended period.

On the other hand, the Optimized SARSA algorithm, as depicted by the orange line, begins with an epsilon of 0.3, which decreases consistently with each episode. This reduction happens in a nonlinear way, which might be either exponential or inverse logarithmic. This will lead to a greater sense of exploration in the beginning of the episodes, which will then transition to a sense of exploitation with a learned policy by the agent, with less and less random exploratory actions. At the end of episode 300, epsilon has decreased to around 0.07.

Optimized SARSA's adaptive epsilon decay systemically addresses the exploration-exploitation problem in the earlier stages of learning, making way for the identification of varied states and transitions. Later, the learning phase shifts towards leveraging the obtained strategies for optimized performance. On the contrary, the fixed epsilon in the original SARSA algorithm might impede the efficiency of the learning process by promoting unwanted exploration and randomness in decisions, even after an optimal strategy has been learned.

From Table 5, epsilon values for the traditional SARSA and optimized SARSA are depicted with the number of episodes. Epsilon for Traditional SARSA is a constant 0.3, whereas it approaches a smaller value for Optimized SARSA with an average of 0.1547 for 300 episodes.

The following graph also helps to confirm the belief that epsilon decay is a critical point in this optimized model and helps to improve the

performance level of the optimized SARSA model in the initial graphs. The transition from the exploratory model to the exploiting model helps to accelerate the rate of learning.

The data obtained shows that the process of hyperparameter optimization increases the performance of the SARSA algorithm remarkably. The converging rate of the optimized agent became faster while learning. Its adaptability has also increased. The whole process demonstrated improvements from the perspective of optimization as well. This is also transparent from the visualization of the paths that the algorithm has learned. Table 6 provides a comparison between the SARSA algorithm and the optimized SARSA algorithm.

This paper verifies that even basic RL algorithms such as SARSA can learn well with an adaptive learning strategy. Such results are very relevant for UAV applications, which require prompt decision-making to respond to dynamic environments, unlike static parameter tuning, which cannot counter the variability or complexity associated with applications like UAV control.

In addition, the 3D grid world configuration has been shown to be a very good and realistic setting for testing our agents. This is mainly because tools for visualizing the results of our experiments, such as 3D animation of flight paths and graphical representation of the learning curves, are of great help in understanding the behaviour of our agents and the effectiveness of our reinforcement learning methods.

The graph showing the reward per episode is indicative of the agents' learning journey. At the start, both Traditional and Optimized SARSA agents were pretty terrible: they hit obstacles frequently and fetched very low (negative) rewards. But the optimized SARSA agent recovered much faster. It showed a steady climb early on and managed to settle near the optimal reward (zero) significantly earlier than the traditional agent. That is to say, it achieved the best behavior much sooner. Additionally, the optimized agent was much steadier; its performance did not jump around as much, suggesting its learning policy was more stable.

In summary, the optimized SARSA learned faster but also with much greater stability courtesy of smart features in the form of adaptive learning rates and careful reward shaping.

Curve Optimized SARSA in Figure 5 declines much more slowly and stabilizes very early, which means that it suffers fewer penalties, such as detours or collisions. In opposition, the traditional SARSA agent keeps accumulating huge negative rewards after hundreds of episodes. Obviously, optimized SARSA is much more efficient and enhances decision-making, most probably due to its decaying epsilon and heuristically guided exploration.

The steps per episode graph shows the efficiency of the agent in navigation. For the initial episodes, the number of steps taken by both agents is large, meaning they were not efficient in exploration. But when the number of episodes grows, the number of steps taken by the optimized SARSA agent is lower than the original agent. It is not only lower but also less variable, meaning more efficient in the application of the learned policy.

Rather, it can be inferred that what makes Optimized SARSA more effective in path planning and execution may be related to reward shaping and early termination constraints that prevent detours and loops.

The figure 8 indicates the optimum performance (minimum number of steps taken)

achieved by the agent until each episode. The optimized SARSA learns not only a lower minimum sooner, but it also improves incrementally throughout the episodes. It finally reaches a point where the optimum number of steps taken is close to being halved compared to the traditional SARSA agent. SARSA learns better optimum paths while navigating because of the effective methods of exploitation, state visitation, and exploration.

From Figure 9, there is an important difference in the modes of exploration of two different algorithms. In traditional SARSA, epsilon is fixed at 0.3 during the entire training phase, whereas in optimized SARSA, epsilon decays. Due to this decay, there is a smooth shift from exploration to exploitation of actions. After reaching the 300th episode, the epsilon value drops to less than 0.07, which helps the agent to leverage the learned policies to their fullest by also minimizing the random actions.

Table 6: Comparison of Traditional and Optimized SARSA

Sno	Metric	Traditional SARSA	Optimized SARSA	Implication
1	Reward per Episode	High variance, slow convergence, unstable learning	52% faster convergence, smoother reward progression	Improved stability and learning efficiency
2	Cumulative Reward	Rapid accumulation of negative rewards, no clear stabilization	Lower accumulation of (68% reduction) negative rewards, early stabilization	Long-term policy efficiency and robustness
3	Steps per Episode	High and inconsistent step count, delayed efficiency	Fewer steps(60% reduction) per episode, consistent and efficient learning	Better navigation strategies and energy efficiency

4	Minimum Steps Achieved So Far	Slower improvement, higher final minimum step count	Rapid (36%) improvement, lower and earlier minimum steps	Early identification of optimal paths
5	Epsilon Decay Over Episodes	Constant epsilon = 0.3, persistent exploration	46.7% reduction in average exploration Decaying epsilon, shifting exploration to exploitation	Smarter exploration control and policy execution

In a wide cross-metric analysis, it is evident that the optimized SARSA algorithm always performs better in complicated navigation tasks. The SARSA algorithm is optimized by making use of significant modifications, such as Adaptive Learning Rate to optimize Q-value parameter updates, Decay Epsilon to manage a seamless transition between exploration and exploitation phases, State Visitation Heuristics to direct the agent towards unexplored regions and thereby prevent trapping in local minima, Reward Shaping to motivate positive actions and avoid inefficient routes towards destinations, and Termination Conditions to avoid paths going through excessively long routes for path termination.

Consequently, the learning process for the optimized agent is faster, and it achieves better policy values, requires fewer steps, and avoids adverse cumulative rewards. Indeed, this is even more critical in practical scenarios for 3D aerial robots, where issues concerning energy efficiency, real-time response, and safety are crucial.

7. CRITIQUES AND LIMITATIONS OF SARSA

SARSA based Reinforcement Learning is more suitable for autonomous aerial robotic exploration and navigation tasks. SARSA is an on policy learning algorithm, where the policy is updated using the actions actually taken including exploratory actions. The challenge in this algorithm is that the convergence is usually slower than off policy algorithms like Q-learning. The algorithm is very sensitive to the used exploration strategy and an inefficient exploration can affect the learning performance a lot. However, in obstacle-rich and dynamic environments, SARSA may prefer safer but longer navigation paths, which may result in suboptimal training performance. In addition, the conventional tabular SARSA has a large state-action space for the 3D UAV environment, which causes the higher memory and computational complexity even after optimization. The algorithm also requires long training episodes for stable learning, posing difficulties for real-time implementation in resource-limited scenarios

8. NEED OF THIS RESEARCH STUDY

The proposed optimized SARSA approach has much better learning efficiency, convergence behavior, adaptability and suitability for complex 3D UAV environments than the existing methods. The state-of-the-art SARSA algorithms heavily rely on fixed learning parameters and simple exploration strategies, leading to slow convergence, poor exploration, and limited adaptability in dense obstacle-rich aerial environments. Most of the previous studies aim to prove the feasibility of SARSA for UAV navigation and obstacle avoidance, without discussing its practical limitations for the real-time autonomous aerial operations.

Most of the earlier works, shown in [1], [2], [8], and [12], shows that the traditional SARSA uses static epsilon-greedy exploration policies, constant learning rates and discrete state-action representations. The studies shows that on-policy learning is more stable and can lead to safer navigation behaviour than some off-policy methods. However, the longer training time, inefficient path selection, inability to adapt quickly to changes in the environment and poor scalability in complex 3D space are the existing challenges.

Also, the conventional SARSA often failed to strike an effective balance between exploration and exploitation, consequently resulting in suboptimal navigation performance and higher collision risks in cluttered environments.

The proposed optimized SARSA framework proposes adaptive optimization mechanisms specifically designed for improving the exploration and navigation performance of UAVs in the obstacle rich 3D environment. The optimized algorithm also incorporates dynamic parameter tuning strategies, such as adaptive epsilon-greedy exploration, variable learning rates, improved reward shaping and enhanced state-space optimisation to overcome the limitations of the existing technology. These

improvements aid the UAV agent to learn faster, make smarter decisions in navigation, and avoid unnecessary exploration while learning.

The optimised SARSA framework aims to address realistic 3D aerial robotic exploration with dense obstacles, dynamic navigation conditions and autonomous path planning requirements against the conventional method which explores only simplified or semi-structured environments. The proposed approach also improves the convergence speed of classical SARSA, in addition to retaining the stability advantages of on-policy learning.

The optimized SARSA model balances, the computational efficiency and intelligent navigation compared with Deep Reinforcement Learning approaches such as DQN and Deep-SARSA presented in [2], [4], [9] and [10]. Deep learning-based methods are highly adaptive, but require large computational resources, memory and training data, which are not feasible for lightweight UAV platforms. On the other hand, the optimized SARSA method is intended to achieve efficient autonomous learning with reduced computational complexity, which is more suitable for real-time embedded UAV applications.

Hence, the main contribution of the proposed research is to develop an optimized SARSA-based reinforcement learning framework that improves the learning convergence, navigation efficiency, exploration-exploitation balance, obstacle avoidance ability, and real-time adaptability in the complex 3D UAV environments with less computational complexity than the deep reinforcement learning approaches. This contribution overcomes the major limitations identified in traditional methods and close the gap between lightweight classical RL methods and computationally expensive DRL techniques.

9. OPEN RESEARCH ISSUES

Although the optimised SARSA-based reinforcement learning for aerial robotic exploration and navigation highlights lot of advantages,a number of unsolved problems and open research challenges still hinder the practical deployment of autonomous UAV systems in real-world 3D environments.

The main issue is that SARSA converges slowly in large, high-dimensional state-action spaces. The optimisation methods improve the learning efficiency, the aerial robotic environments still require heavy computation because of the continuous interaction of UAVs with dynamic obstacles, different altitudes and uncertain environmental conditions. The increase of the environment complexity leads to the exponential growth of state-action combinations, which results in delayed policy convergence and lengthy training time.

Another important factor is the exploration-exploitation trade-off. Optimised SARSA attempts to balance exploration of the environment and exploitation of the learned policy using adaptive exploration strategies such as epsilon decay or reward shaping. However, in the case of complex aerial missions, too much exploration may lead to unsafe navigation and collisions, while too little exploration may trap the UAV in locally optimal paths. Designing safe and efficient adaptive exploration mechanisms in unknown 3D environments remains an open research challenge.

Partial observability of real environments is another important limitation. Most SARSA optimisation studies assume the UAV has perfect environmental perception, using sensors or simulation data. However, in practice, UAV sensors may be affected by noise, occlusion, communication delays or limited sensing range. These uncertainties degrade the navigation reliability and may result in unstable learning behaviour, inaccurate state estimation and poor obstacle avoidance performance.

Another open challenge is the scalability of optimised SARSA on continuous 3D navigation spaces. The classic SARSA and many optimised variants are typically intended for discrete state representations. The real aerial environments are continuous and highly dynamic. Hence, efficient discretisation or function approximation techniques are required. The discretisation being poor will increase memory usage and decrease the optimality of the path, while the complex approximators could increase the on-board computational load.

Considering the onboard real time implementation, the UAV platforms typically have limited processing power, memory and battery energy. Optimised SARSA is less computationally intensive than deep reinforcement learning approaches. However, many optimisation mechanisms such as adaptive parameter tuning, prioritised learning, hybrid heuristics or multi-agent coordination add additional computational

complexity. Fast learning and decision making under tight hardware constraints is still considered to be a major constraint.

Another research problem is the design of reward functions. UAV navigation tasks often involve multiple objectives, such as shortest-path generation, obstacle avoidance, energy efficiency, target discovery, flight stability, and mission completion time. It is difficult to design a reward function that meets all these objectives simultaneously without leading to unstable learning or reward conflicts. Poorly designed reward can cause the robot to oscillate during navigation, explore inefficiently, or take unsafe actions.

Another unresolved issue is the generalisation capability of optimised SARSA models. Most of the existing studies are based on the evaluation of the UAV performance in a simulated environment with a predefined set of obstacles. The policies learned in one environment often do not transfer to new or real-world scenarios, due to environmental uncertainty, weather changes, sensor variability and dynamic obstacles. Simulation-to-reality gap bridging is still a large open problem in aerial robotics.

Multi-UAV coordination increases the complexity. Most of the optimised SARSA studies are based on the single agent UAV navigation, while the practical disaster response and exploration missions have more and more requirements for the cooperative multi-agent systems. Challenges in extending optimised SARSA to multi-agent environments include distributed learning, communication overhead, collision avoidance among agents, cooperative reward allocation and convergence stability.

Nextly, the autonomous exploration with energy-awareness is a major problem. Most of the existing approaches are for navigation efficiency and obstacle avoidance, with little attention paid to battery-aware decision making, power optimisation and flight endurance. The operation time of the UAV is greatly limited by the battery capacity, thus incorporating energy-efficient learning in SARSA optimisation is still essential for long-term missions.

Also, the robustness of optimised SARSA in dynamic and unpredictable environments is not sufficiently explored. In the real world, aerial missions may involve moving obstacles, smoke, debris, wind disturbances, GPS denial or communication failure. The existing optimised SARSA schemes cannot adapt sufficiently fast to the rapidly changing environments, especially in the online learning mode in real-time.

Finding optimal adaptive values for learning rate α, discount factor γ and exploration parameters is an open optimisation problem in highly dynamic aerial environments which directly impacts convergence stability and navigation performance.

The open research direction is the development of a lightweight, adaptive, scalable, energy-efficient, and real-time optimised SARSA framework that is capable of robust autonomous exploration and navigation in uncertain 3D aerial environments with low computational overhead for practical UAV deployment.

10. CONCLUSION AND FUTURE WORK

In summary, based on this research work it is proved that the learning parameter adaptation greatly increases the agent's ability to explore a complex 3D environment. This research also discusses the value of combining optimization approaches with the creation of reinforcement learning systems for aerial robotics. The simulated results shows that optimized SARSA gives a 55% improvement in overall efficiency, 52% faster convergence and 70% more stability as compared to traditional SARSA. Future research may focus on merging SARSA with deep learning for function approximation, developing model-based reinforcement learning for more efficient exploration or testing the trained agents on real-life physical UAV platforms. Moreover, it might also help to extend this work for handling dynamic obstacles and multi-agents. Based on the results of the proposed five comparison scenarios, it is proved that the optimized SARSA learning algorithm shows a better performance in terms of speed, efficiency during navigation. Hence the need of optimization for this research is proved to be essential.

REFERENCES

[1] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2020,

[2] Yalew Zelalem Jembre , Yuniarto Wimbo Nugroho, Muhammad Toaha Raza Khan, Muhammad Attique, Rajib Paul, Syed Hassan Ahmed Shah and Beomjoon Kim, Evaluation of Reinforcement and Deep Learning Algorithms in Controlling

Unmanned Aerial Vehicles., MDPI, Applied sciences, 2021.

[3] Divija Swetha Gadiraju, Prasenjit Karmakar, Vijay K. Shah and Vaneet Aggarwal, GLIDE Multi-Agent Deep Reinforcement Learning for Coordinated UAV Control in Dynamic Military Environments, MDPI, Information, 2024

[4] Gianluca Fontanesi, Anding Zhu, Mahnaz Arvaneh, , & Hamed Ahmadi, A Transfer Learning Approach for UAV Path Design with Connectivity Outage Constraint, IEEE, Internet of Things, 2021.

[5] Wenlong Meng1, Xuegang Zhang, Lvzhuoyu Zhou1, Hangyu Guo1 and XinHu , Advances in UAV Path Planning: A Comprehensive Review of Methods, Challenges, and Future Directions, MDPI, Drones, 2025

[6] Raúl Arranz, David Carramiñana, Gonzalo de Miguel, Juan A. Besada & Ana M. Bernardos Application of Deep Reinforcement Learning to UAV Swarming for Ground Surveillance, MDPI, Sensors, 2023.

[7] Fadi AlMahamid, Katarina Grolinger, Autonomous Unmanned Aerial Vehicle Navigation using Reinforcement Learning: A Systematic Review, Elsivier, 2022, Volume 115, Issue C, Pg1–21. https://doi.org/10.1016/j.engappai.2022.105321

[8] Mohamad Hafiz Abu Bakar, Abu Ubaidah Shamsudin1, Ruzairi Abdul Rahim, Zubair Adil Soomro, Andi Adrianshah, Comparison Method Q-Learning and SARSA for Simulation of Drone Controller using Reinforcement Learning, Journal of Advanced Research in Applied Sciences and Engineering Technology, 2023, 30, Issue 369-78

[9] Kangtong Mo, Linyue Chu, Xingyu Zhang, Xiran Su, Yang Qian, Yining Ou & Wian Pretorius, DRAL: Deep Reinforcement Adaptive Learning for Multi-UAVs Navigation in Unknown Indoor Environment, 2024 arXiv:2409.03930v1 [cs.RO] 5 Sep 2024

[10] Wei Luo, Qirong Tang, Changhong Fu, & Peter Eberhard, Deep-Sarsa based Multi-UAV Path Planning and Obstacle Avoidance in a Dynamic Environment, Chapter in Lecture Notes in Computer Science, (2018) DOI: 10.1007/978-3-319-93818-9_10 · June 2018.,

[11] Priyanka Sarma, Atowar Ul Islam, Implementation of Hybrid Adaptive Learning Algorithm for Task Offloading, Indian Journal of Science and Technology, (2024) 17(38): 3929-3936. https://doi.org/10.17485/IJST/v17i38.2280

[12] Guan-Ting Tu & Jih-Gau Juang, UAV Path Planning and Obstacle Avoidance Based on Reinforcement Learning in 3D Environments, MDPI, Actuators, (2014)

[13] Shah Asif Bashir, Dr. Farida Khursheed, Dr. Ibrahim Abdoulahi, Adaptive-Greedy Exploration for Finite Systems, GEDRAG & ORGANISATIE REVIEW - ISSN:0921-5077, (2021), VOLUME 34: ISSUE 04, page 417- 431

[14] Jieun Park, Dokkyun Yi and Sangmin Ji, (2020), A Novel Learning Rate Schedule in Optimization for Neural Networks and It's Convergence, Symmetry 2020, 12, 660; doi:10.3390/sym12040660