Multi-Target Path Planning with Probabilistic Detection in Cluttered Environments

Noor Khial[1], Naram Mhaisen[2], Loay Ismail[1], Mohamed Mabrok[3], Amr Mohamed[1]
[1] College of Engineering, Qatar University, Qatar
[2] College of Electrical Engineering, Mathematics, and Computer Science, TU Delft, The Netherlands
[3] College of Arts and Sciences, Qatar University, Qatar
Email: {noor.khial, loay.ismail, m.a.mabrok, amrm}@qu.edu.qa, N.Mhaisen@tudelft.nl

ICC 2025 - IEEE International Conference on Communications | 979-8-3315-0521-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICC52391.2025.11161388

***Abstract*—Autonomous Unmanned Aerial Vehicles (UAVs) offer substantial advantages for tasks such as surveillance, disaster management, and environmental monitoring, where human intervention can be risky. With advancements in their agility and autonomy, UAVs are becoming essential for critical tasks in combat, reconnaissance, wildfire monitoring, and disaster search and rescue. This paper addresses a key challenge in UAV path planning: efficiently visiting multiple unknown mobile targets in complex, obstacle-filled environments. We leverage the Deep Deterministic Policy Gradient (DDPG) framework to continuously control UAV movement to enable effective obstacle avoidance and sequential target visitation. Our approach allows the UAV to learn the unknown distribution of mobile targets and determine optimal paths while navigating around obstacles. With limited environment information, the agent receives rewards based on the confidence of detecting targets within its observation field. We validate the effectiveness of our method through comparison with an optimal benchmark that assumes perfect knowledge of target mobility and obstacle locations. Results indicate that increasing target numbers significantly impacts the agent's performance by requiring additional training time. Moreover, heavily cluttered environments reduce mission success rates for target visitation.**

***Index Terms*—UAV, path planning, mobile targets, partial observability, Reinforcement Learning.**

I. Introduction

Autonomous Unmanned Aerial Vehicles (UAVs) have been used for a variety of applications, such as environmental monitoring, surveillance, precision agriculture, and urban planning. UAVs are capable of navigating challenging operational environments, such as disaster zones, dense urban landscapes, forests, and other cluttered terrains, where traditional methods may face limitations. Their agility, speed, and autonomous capabilities make UAVs well-suited for tasks that are impractical or unsafe for human operators.

A critical task for UAVs in dynamic environments is the visitation of multiple targets, particularly when these targets are mobile and their movement patterns are unknown. These environments are often cluttered with obstacles that complicate navigation. While path planning in unknown environments has been extensively studied using techniques such as Markov Decision Process (MDP), many existing approaches focus only on single target visitation. Studies employing techniques like reinforcement learning (RL) and the A* algorithm [1], [2] have demonstrated effective obstacle avoidance, but they typically don't address the complexity of visiting multiple mobile targets while avoiding obstacles.

To address the challenge of target visibility in cluttered environments, where obstacles can obstruct the line of sight, we introduce a model for observing targets that uses a confidence interval approach. Our model incorporates probabilistic sensing modalities (e.g., LiDAR, radar). Unlike previous work on search and target visitation, such as in [3], [4], which relied on a binary observation model, 0 if the target was not observed, or 1 if it was. Our model assigns a probabilistic value to each target observation. This probabilistic approach, similar to frameworks used in studies like [5], better accounts for uncertainties in complex environments. By representing observations as confidence intervals, our model captures the uncertainty inherent in target detection, adapting to fluctuations in visibility common in cluttered settings.

To handle both the uncertainty of target movements and the complexity of obstacle-rich environments, we leverage an MDP-based RL approach. This method enables the UAV (agent) to learn the unknown mobility pattern of targets and find the optimal path to visit them. While RL shows potential in UAV systems, most existing studies focus on general area scanning and coverage [6], [7] or involve continuous engagement with targets [8], [9]. Our work uniquely focuses on detecting and visiting multiple targets sequentially without assuming continuous interaction, better reflecting real-world scenarios where targets are mobile with unknown patterns.

Some studies have explored RL to adapt to incomplete or changing information with partially observable targets [10], [11], but these approaches often assume predictable target movement patterns. In our approach, the UAV can only observe a portion of the environment. As targets move in and out of the observable area with unknown patterns, the UAV faces a dynamic decision-making challenge of planning paths with only local information about target locations.

Our key contributions include:

- Formulating the multiple-target visitation problem in

Research reported in this publication was supported by the Qatar Research Development and Innovation Council ARG01-0527-230356. The content is solely the responsibility of the authors and does not necessarily represent the official views of Qatar Research Development and Innovation Council.

obstacle-filled environments as an optimization problem, with constraints on the UAV's movements and consideration of its limited visibility of the environment.
- Apply RL techniques for optimal path planning to visit all mobile targets and avoid obstacles, assuming the targets' mobility is unknown.
- Conduct experimental evaluation of the RL agent's adaptability across various scenarios such as navigating in different clutter levels and compare it with a benchmark.

The remainder of this paper is organized as follows: Section II presents our system model and problem formulation. Section III introduces the reinforcement learning approach used to address the problem. Finally, Section IV presents the experimental results with a discussion of our findings.

II. System Model & Problem Formulation

In this section, we describe the system model illustrated in Fig. 1, where a UAV is tasked with searching for multiple mobile targets in a cluttered environment.

Environment. The environment is modeled as a plane of two dimensions, continuous, and bounded area. Any point within the environment is represented as $(x, y) \in \mathbb{R}^2_+ = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x \leq X,\ 0 \leq y \leq Y\}$, with X and Y representing the boundaries of $\mathcal{M}$. There exists a set of stationary obstacles $\mathcal{O} = \{o_n, \forall n \in [O]\}$, where each obstacle o_n has a rigid shape but may vary in size.

Targets. A starting area $\boldsymbol{\mathcal{S}_0}$ is defined where the mobile targets begin their movement, with the goal of reaching a targeted zone $\boldsymbol{\mathcal{S}_f} \in \mathbb{R}^2_+$. The mobile targets, denoted as $\mathcal{H} = \{h_i, \forall i \in [H]\}$, move in formation of a single leader and a number of followers. The leader moves according to $\mathbf{d}^{(t+1)} = \mathbf{d}^{(t)} + \alpha(\mathbf{e}^{(t)} - \mathbf{d}^{(t)})$, where α is the step size and $\mathbf{e}^{(t)} \in \boldsymbol{\mathcal{S}_f}$. Each follower updates its position based on $\mathbf{h}_i^{(t+1)} = \mathbf{h}_i^{(t)} + \beta(\mathbf{d}^{(t+1)} - \mathbf{h}_i^{(t)}), \quad \mathcal{H} = \{h_i, \forall i \in [H]\}$, where β is the follower step size. The trajectory of $\mathcal{H}$ is simulated using a Bézier curve [12] for collision avoidance, expressed as $B^{(t)}(h_i) = \sum_{k=0}^{n} \binom{n}{k}(1-t)^{n-k}t^k c_{i,k}, \quad \mathcal{H} = \{h_i, \forall i \in [H]\}$, where n is the curve degree. If no obstacle is found, h_i will follow the shortest path from $\boldsymbol{\mathcal{S}_0}$ to $\boldsymbol{\mathcal{S}_f}$.

Action Space for Agent. The UAV operates at a fixed altitude and navigates through actions at discrete time steps $t \in [T]$. The action is represented by $\boldsymbol{a}^{(t)} = [l^{(t)}, \theta^{(t)}]$, where $l^{(t)} \in [0, L]$ defines the step size and $\theta^{(t)} \in [0, 2\pi)$ specifies the direction. The action space is defined as:

$$\mathcal{A} = \{[l, \theta] \mid l \in [0, L], \theta \in [0, 2\pi)\} \tag{1}$$

State Space. The action $\boldsymbol{a}^{(t)}$ transitions the UAV between states $\mathbf{s}^{(t)}$. Due to its limited field of observation (FO) and the unknown mobility of the targets, the UAV does not have access to the complete state. The state space $\mathcal{S}$ includes information about the UAV's position, target positions, and obstacles:

$$\mathcal{S} = \{(\mathbf{p}, \mathcal{H}, \mathcal{O}) \mid \mathbf{p} \in \mathcal{M}, \mathcal{H} \subseteq \mathcal{M}, \mathcal{O} \subseteq \mathcal{M}\} \tag{2}$$

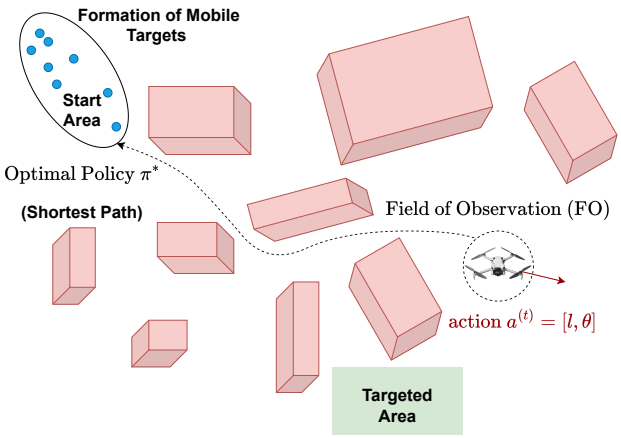


Fig. 1: System Model: Top-View of the UAV Environment.

Observation of the Agent. The agent observes targets and obstacles within its field of observation (FO), represented by:

$$\mathbf{O}^{(t)} = [\mathbf{p}^{(t)}, t, [D^{(t)}(h_i)], [O_n^{(t)}]], \tag{3}$$

where $\mathbf{O}^{(t)} \subseteq \mathbf{s}^{(t)}$. Here, $\mathbf{p}^{(t)} = (x^{(t)}, y^{(t)})$ denotes the UAV location, $D_i^{(t)}(\cdot)$ represents the observation of the i-th target, and $O(\cdot)$ indicates obstacle observations. The optimal path π^* is defined as $\mathcal{P} = \{\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \ldots, \mathbf{p}^{(T)}\}$. The UAV has a limited field of observation (FO) with a radius r and possesses no prior knowledge outside this area. The observation of each target h_i is expressed as a confidence interval given by $D^{(t)}(h_i) = k \cdot ||\mathbf{h}_i^{(t)} - \mathbf{p}^{(t)}||^{-4}$, where the confidence interval is influenced by the distance between the agent and the target, $||\mathbf{h}_i^{(t)} - \mathbf{p}^{(t)}||$, and k is a constant. The agent is capable of observing only the targets that fall within its field of observation (FO) and adjusts its path $\mathcal{P}$ accordingly, with no detection capability outside the FO. Fig. 2 illustrates the confidence interval of a single target in relation to its distance from the agent.

A. Problem Formulation

The objective is to find the shortest path π^* for the agent to visits all the mobile targets $\mathcal{H}$ while avoiding obstacles

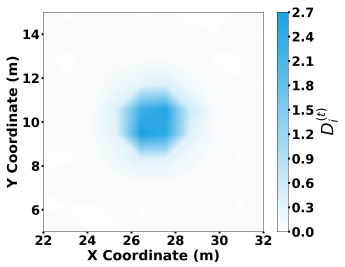


Fig. 2: This map shows the $D^{(t)}(h_i)$ based on the distance between a target and the agent at $(27, 10)$, illustrating how confidence strength varies with distance.

$\mathcal{O}$. Considering $\mathcal{P} = \{\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \ldots, \mathbf{p}^{T}\}$ is the sequence of positions that the UAV will take during its mission, each target $h_i \in \mathcal{H}$ must be visited at least once according to $\sum_{t=1}^{T} \delta(\mathbf{p}^{(t)}, \mathbf{h}_i^{(t)}) \geq 1, \quad \mathcal{H} = \{h_i, \forall i \in [H]\}$. Where $\delta(\mathbf{p}^{(t)}, \mathbf{h}_i^{(t)})$ is an indicator function that equals 1 if the agent detects the presence of a target h_i at time t within the FO, and 0 otherwise. The agent must avoid intersecting with any point within the area of the obstacle for all obstacles $\mathbf{p}^{(t)} \notin \bigcup_{o=1}^{O} \mathcal{L}_o, \quad \forall t \in [T]$.

The agent's movement is constrained by its maximum speed $v_{\max}$, which is controlled by the maximum step size L the agent can take according to $\|\mathbf{p}^{(t+1)} - \mathbf{p}^{(t)}\| \leq v_{\max} \cdot \Delta t$. The confidence strength $D^{(t)}(h_i)$ is subject to a minimum threshold, as $D^{(t)}(h_i) \geq D_{\min}, \quad \forall h_i \in \mathcal{H}, \forall t \in [T]$. This constraint ensures that a target is only considered detected if it is within the UAV's field of observation (FO) and the confidence strength $D^{(t)}(h_i)$ exceeds a minimum threshold $D_{\min}$. The trajectory $\mathcal{P}$ should be continuous and feasible within the environment as $\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} + \boldsymbol{a}^{(t)}, \quad \forall t \in [T]$. Where $\boldsymbol{a}^{(t)}$ is the action vector representing the agent's movement at time t. Therefore, the final optimization problem:

$$\mathbf{P}: \min_{\mathcal{P}} \quad \sum_{t=1}^{\mathcal{T}-1} ||\mathbf{p}^{(t+1)} - \mathbf{p}^{(t)}|| \tag{4a}$$

Subject to:

$$\sum_{t=1}^{\mathcal{T}-1} \delta(\mathbf{p}^{(t)}, \mathbf{h}_i^{(t)}, D^{(t)}(h_i)) \geq 1,$$
$$\mathcal{H} = \{h_i, \forall i \in [H]\}, \tag{4b}$$
$$\mathbf{p}^{(t)} \notin \bigcup_{o=1}^{O} \mathcal{L}_o, \qquad \forall t \in [T], \tag{4c}$$
$$||\mathbf{p}^{(t+1)} - \mathbf{p}^{(t)}|| \leq v_{\max} \cdot \Delta t, \qquad \forall t \in [T], \tag{4d}$$
$$D^{(t)}(h_i) \geq D_{\min},$$
$$\mathcal{H} = \{h_i, \forall i \in [H]\}, \forall t \in [T], \tag{4e}$$
$$\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} + \boldsymbol{a}^{(t)}, \qquad \forall t \in [T]. \tag{4f}$$

The final trajectory $\mathcal{P}$ ensures that all targets are visited while avoiding obstacles in the environment. However, solving the problem **P** directly using optimization techniques is infeasible because we assume that $\mathbf{h}_i^{(t)}$ and $D^{(t)}(h_i)$ are unknown a prior. Therefore, we apply RL technique to solve the optimization problem. The RL agent can learn the transition probabilities from one state to another and predict the next position $\mathbf{p}^{(t+1)}$ to find the optimal path.

III. Proposed Solution

To address the path planning problem with multiple targets, as described in Eq. (4), we utilize the Deep Deterministic Policy Gradient (DDPG) algorithm, an actor-critic method designed for continuous action spaces.

DDPG Agent. DDPG is a model-free, off-policy reinforcement learning algorithm that is designed for continuous action spaces. It was first introduced in [13]. The algorithm combines the benefits of both Deep Q-Learning (DQN) and policy gradient methods within an actor-critic framework.

Multiple studies have used DDPG for path planning, such as [14]–[18]. These studies discussed several important features of DDPG, including its capability to navigate dynamic environments, adapt to real-time changes, and coordinate multi-agent systems for collision-free navigation. Additionally, DDPG has been noted for reducing computation times relative to traditional optimization methods [13], [14].

A. DDPG Model

The DDPG model comprises two components: an actor model $\pi(\mathbf{O}|\theta^{\pi})$ and a critic model $Q(\mathbf{O}, \mathbf{a}|\theta^{Q})$. The actor network outputs a deterministic policy mapping observations $\mathbf{O}$ to optimal actions $\mathbf{a}$, while the critic network estimates the Q-value for each state-action pair.

Actor Network. The actor network $\pi(\mathbf{O}^{(t)}|\sigma^{\pi})$ is parameterized by σ^{π} and outputs a deterministic policy:

$$\mathbf{a}^{(t)} = \pi(\mathbf{O}^{(t)}|\sigma^{\pi}) + N^{(t)} \tag{5}$$

where $N^{(t)}$ is exploration noise, typically implemented as an Ornstein-Uhlenbeck process [13]. However, in our scenario, we adopted a random exploration that is decaying exponentially during the training phase.

Critic Network. The critic network $Q(\mathbf{O}, \mathbf{a}|\sigma^{Q})$, parameterized by σ^{Q}, estimates the Q-value:

$$Q(\mathbf{O}^{(t)}, \mathbf{a}^{(t)}|\sigma^{Q}) \approx \tilde{R}^{(t)}(\cdot) + \gamma Q(\mathbf{O}^{(t+1)}, \pi(\mathbf{O}^{(t+1)}|\sigma^{\pi})|\sigma^{Q}) \tag{6}$$

The term $\tilde{R}^{(t)}$ is a function that outputs the actual reward received at time t after executing action $\mathbf{a}^{(t)}$, according to the observation $\mathbf{O}^{(t)}$. The discount factor γ adjusts the weight of future rewards compared to immediate rewards.

B. Exploration Process

The DDPG algorithm uses experience replay and soft target updates to stabilize training. The critic is updated by minimizing the loss:

$$w = \mathbb{E}[(\tilde{R}^{(t)}(\cdot) + \gamma Q'(\mathbf{O}^{(t+1)}, \pi'(\mathbf{O}^{(t+1)}|\sigma^{\pi'})|\sigma^{Q'}) - Q(\mathbf{O}^{(t)}, \mathbf{a}^{(t)}|\sigma^{Q}))^2] \tag{7}$$

The loss w is computed as the difference between the target Q-value and the Q-value estimated by the current critic network, $Q(\mathbf{O}^{(t)}, \mathbf{a}^{(t)}|\sigma^{Q})$, and is minimized during training to refine the critic network's accuracy in estimating expected rewards. Q' and π' are target networks. The actor is updated using the deterministic policy gradient theorem [13]:

$$\nabla_{\sigma^{\pi}} J \approx \mathbb{E}[\nabla_{\mathbf{a}} Q(\mathbf{O}, \mathbf{a}|\sigma^{Q})|_{\mathbf{O}=\mathbf{O}^{(t)}, \mathbf{a}=\pi(\mathbf{O}^{(t)})} \nabla_{\sigma^{\pi}} \pi(\mathbf{O}|\sigma^{\pi})|_{\mathbf{O}=\mathbf{O}^{(t)}}] \tag{8}$$

The DDPG algorithm aims to gradually approach the optimal policy π^*, by iteratively improving $\pi(\mathbf{O}|\sigma^{\pi})$ policy. This

is achieved by following the policy gradient $\nabla_{\sigma^\pi} J(\pi)$, which indicates how to adjust the policy parameters σ^π to maximize the expected return $J(\pi)$. Target networks are updated softly:

$$\sigma^{Q'} \leftarrow \tau\sigma^{Q} + (1-\tau)\sigma^{Q'} \tag{9}$$

$$\sigma^{\pi'} \leftarrow \tau\sigma^{\pi} + (1-\tau)\sigma^{\pi'} \tag{10}$$

C. Adaptation for Multi-Target Path Planning

Recall the definition of the state, the action space definition, and the observation of the agent, the reward is computed only for non-visited targets ($\mathcal{H}_f \in \mathcal{H}$). We define the reward function to solve the problem in Eq. (4) as follows:

Reward Function. During operation, the agent observes confidence strength $D^{(t)}(h_i)$ based on its current position $\mathbf{p}^{(t)}$ relative to the target's position. The confidence strength is non-zero when the target is within the agent's Field of Observation (FO), and approximately zero otherwise. The reward of the agent is defined in terms of the confidence reward as:

$$R^{(t)}(\mathbf{p}^{(t)}, \{\mathbf{h}_i^{(t)}\}) = \sum_{h_i \in \{\mathcal{H}\}\setminus\{\mathcal{H}_f\}} - \left[D^{(t)}(0) - D^{(t)}(h_i, \mathbf{p}^{(t)} - \mathbf{h}_i^{(t)})\right] \tag{11}$$

where $D^{(t)}(0)$ represents the maximum confidence strength achievable when the distance between the agent $\mathbf{p}$ and target $\mathbf{h}_i$ is zero. For each non-visited target, the agent calculates the cumulative $D^{(t)}(h_i)$ when the target is within the FO, relative to the maximum possible confidence. This reward structure penalizes the agent for each time step it fails to detect a target while encouraging it to minimize distance to targets.

$$\tilde{R}^{(t)} = \begin{cases} R^{(t)}(\mathbf{p}^{(t)}, \{\mathbf{h}_i^{(t)}\}), & \text{if } \mathbf{p}^{(t)} \notin \bigcup_{o=1}^{O} \mathcal{L}_o \\ & \mathbf{p}^{(t)} \in \mathcal{M} \ \&\ \mathbf{h}_i^{(t)} \in \text{FO} \\ -\chi, & \text{if } \mathbf{p}^{(t)} \notin \bigcup_{o=1}^{O} \mathcal{L}_o \\ -2\cdot\chi, & \text{if } \mathbf{p}^{(t)} \text{ is near } \partial\mathcal{M} \end{cases} \tag{12}$$

Where χ is a penalty value given to the agent. Furthermore, the agent is penalized if it approaches or intersects with the boundaries of the environment, represented by the limits $x = 0$, $x = X$, $y = 0$, and $y = Y$. This boundary penalty ensures that the agent does not become stuck at the edges of the $\mathcal{M} \subseteq \mathbb{R}^2$, encouraging it to explore the entire environment. The objective of the agent is to reach the optimal policy π^*, which maximizes the reward $\tilde{R}$. The reward function $\tilde{R}$ is designed to define the shortest path that allows the agent to visit all targets in the shortest possible time while avoiding stationary obstacles $\mathcal{O}$. Algorithm 1 provides a detailed, step-by-step implementation of the DDPG algorithm.

Algorithm 1 DDPG Algorithm

Input: Observation $\mathbf{O}$, actor $\pi(\mathbf{O}|\sigma^\pi)$, critic $Q(\mathbf{O}, \mathbf{a}|\sigma^Q)$
Initialize π and Q with random weights, target networks π' and Q' with same weights, and replay buffer
for each episode **do**
    Initialize exploration and get initial observation $\mathbf{O}^{(0)}$
    for each time step t **do**
        Select action $\boldsymbol{a}^{(t)}$, execute it, observe reward $\tilde{R}$ and new observation $\mathbf{O}^{(t+1)}$
        Store $(\mathbf{O}^{(t)}, \boldsymbol{a}^{(t)}, \tilde{R}, \mathbf{O}^{(t+1)})$ in replay buffer
        Sample mini-batch from buffer, update critic and actor, and soft-update target networks
    end for
end for

IV. PERFORMANCE EVALUATION

This work aims to enable a UAV to autonomously discover and visit unknown targets in complex, cluttered environments while avoiding obstacles. We focus on a RL approach for exploring and detecting targets through walls without prior knowledge of target locations. Our evaluation examines the agent's adaptability to varying target numbers and environmental clutter, expecting near-benchmark performance with consistent target detection and effective obstacle avoidance.

Experimental Setup. Table I shows the configurations for the environment and the DDPG model hyperparameters.

Environment. The environment size is defined as $\mathcal{M}(X, Y) = (0, 55)$ with action space $\mathcal{A} = \{[l, \theta] \mid l \in [0, 5], \theta \in [0, 2\pi)\} \subseteq \mathbb{R}^2$. The minimum detection strength D_{min} for marking a target as *visited* is 2, with rewards calculated using Eq. 11. The obstacle/boundary penalty χ is 400, movement step sizes α and β are 0.2, and starting area $\boldsymbol{\mathcal{S}_f}$ is $\{(x, y) \mid x \in [0, 6], y \in [0, 6]\}$.

DDPG Model. The critic and actor learning rates are 10^{-3} and 10^{-4} respectively, with discount factor $\gamma = 0.99$ and target update rate $\tau = 0.005$. The experience replay buffer size is 20,000, batch size is 64.

Benchmark. We implement the Dijkstra's algorithm under the assumption that the locations of all targets $h_i \in \mathcal{H}$ are known *a priori*. Although this assumption is unrealistic in practical scenarios, where targets are initially unknown and must be discovered through exploration, this idealized case provides a desirable benchmark. It serves as an upper bound on performance, against which we can evaluate more practical approaches that operate with incomplete information.

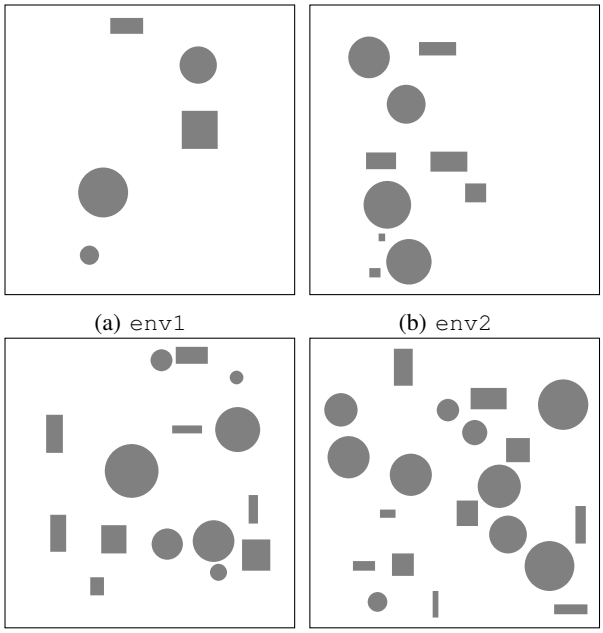
Evaluation Metric. We evaluate the RL agent's performance by showing the cumulative reward convergence during training. In inference, we measure success rate (targets covered), path length, and obstacle hit rate. The DDPG agent's inference performance is averaged over 20 episodes and compared to the Dijkstra's algorithm.

A. Impact of clutter level on the agent's performance

To assess the effect of clutter on agent performance, we created four environments with increasing clutter, as shown

TABLE I: Parameters for the environment and models.

Environment		DDPG Model	
Parameter	**Value**	**Parameter**	**Value**
$\mathcal{M}(X, Y)$	$(0, 55)$	Critic learning rate	10^{-3}
θ	$[0, 2\pi]$	Actor learning rate	10^{-4}
L	$[0, 5]$	γ	0.99
$\boldsymbol{\mathcal{S}_f}$	$\{x, y \in [0, 6]\}$	τ	0.005
D_{min}	2	Buffer size	20000
χ	400	Batch size	64
α, β	0.2	Time Steps	200



(a) `env1` (b) `env2` (c) `env3` (d) `env4`

Fig. 3: The environments were designed to test the impact of clutter levels on the agent's performance. Each environment features a different level of clutter, determined by the number of obstacles present.

in Fig. 3. Each environment contains randomly placed, non-overlapping circles and rectangles, with obstacle size and position set randomly within boundaries. Obstacle count increases with clutter level.

Discussion. Fig. 4 illustrates the agent's training performance, demonstrating that increased clutter in the environment adds complexity to the target visitation mission. As the agent navigates around obstacles on its way to targets, it encounters more frequent collisions. Table II presents the DDPG agent's inference performance compared with the benchmark model. The DDPG agent maintains a high success rate (e.g., 100% for `env1` and 95% for `env2`, `env3`, and `env4`), although increased clutter impacts path efficiency. In the more complex environments, such as `env3` and `env4`, the DDPG agent travels significantly longer paths compared to the benchmark to avoid obstacles. For instance, in `env4`, the DDPG agent covers 43.62 units compared to the benchmark's 25.14 units while successfully avoiding obstacles.The time steps required

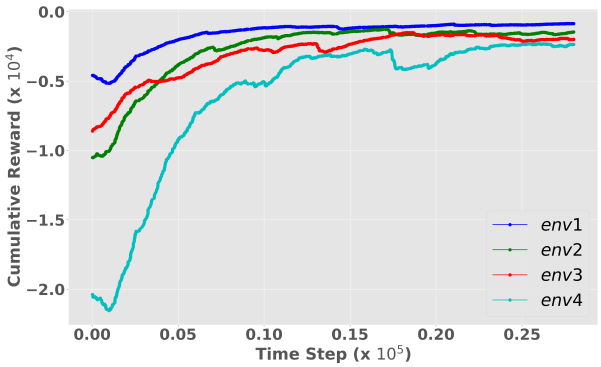


Fig. 4: DDPG agent performance in training under different clutter levels.

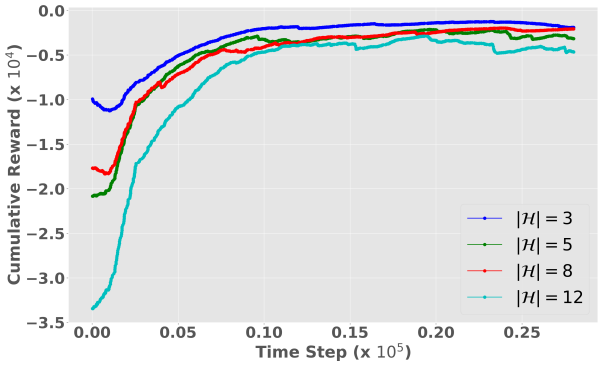


Fig. 5: DDPG model training with different number of targets.

to reach the targets highlight the agent's approach to planning: in `env4`, the DDPG agent completes the task in 14 steps, while the benchmark requires 19 steps. This difference indicates that the DDPG agent is more efficient in terms of time steps, though it takes longer paths to prioritize Obstacle avoidance.

B. Impact of number of targets over the performance of the agent

In this experiment, we assess the agent's ability to visit more targets within `env3` (see Fig. 3c), starting from $\mathcal{S}_0$ (Table I). The agent's performance is evaluated with 3, 5, 8, and 12 targets, each with end points $\mathbf{e}^{(t)}$ sampled from a normal distribution $\mathcal{N}((55, 55), 0.8^2)$.

Discussion. As the number of targets increases, the agent's performance is impacted, as shown in Fig. 5. More targets add complexity, affecting convergence during training. As the agent needs more time for exploring and visiting more number of targets. During inference, the DDPG agent's success rate drops slightly with more targets, from 95% with 3 targets to 86.7% with 12 targets, while the benchmark maintains a 100% success rate.

TABLE II: Performance of the DDPG agent during inference with varying levels of clutter.

Environment	Success Rate %		Path Total Distance		Time Steps		Obstacle Hit Rate %	
	DDPG	Benchmark	DDPG	Benchmark	DDPG	Benchmark	DDPG	Benchmark
env1	100	100	26.01	24.56	6	17	0	0
env2	95	100	32.41	26.97	7	18	0	0
env3	95	100	31.63	26.56	9	15	0	0
env4	95	100	43.62	25.14	14	19	0	0

TABLE III: Inference performance of the DDPG agent under varying numbers of targets.

Number of Targets	Success Rate %		Path Total Distance		Time Steps		Obstacle Hit Rate %	
	DDPG	Benchmark	DDPG	Benchmark	DDPG	Benchmark	DDPG	Benchmark
3	95	100	30.6	23.56	19	19	0.1	0
5	90	100	42.2	24.6	20	20	0.2	0
8	90	100	44.72	27.38	24	25	0.3	0
12	86.7	100	45.89	29.38	29	27	0.37	0

Path distance and time steps also increase with target count: from 30.6 units and 19 steps for 3 targets to 45.89 units and 29 steps for 12 targets. Although comparable in steps, the benchmark favors the shortest path. The DDPG agent shows a minor rise in Obstacle hit rate, from 0.1 events for 3 targets to 0.37 for 12 targets, with none for the benchmark.

These results highlight the DDPG agent's trade-offs in balancing target visitation and Obstacle hit rate, with slight performance declines as targets increase.

V. Conclusion

This work introduces the application of the DDPG model to optimize path planning for UAVs tasked with multi-target visitation in continuous and cluttered environments. To address this problem, the DDPG model was applied under the assumption of stationary obstacles and stochastic mobility pattern for targets. The results show the feasibility of the proposed solution achieving performance near the benchmark, which assumes perfect knowledge about the environment. However, this work has not accounted for the dynamic nature of obstacles, which is an inherent feature in tasks such as search and rescue operations. Therefore, it is essential to consider both the mobility of unknown targets and the changing dynamics of obstacles.

References

[1] L. G. S. da Rocha, K. A. Q. Caldas, M. H. Terra, F. Ramos, and K. C. T. Vivaldini, "Dynamic q-planning for online uav path planning in unknown and complex environments," *arXiv preprint arXiv:2402.06297*, 2024.

[2] Q. Ren, Y. Yao, G. Yang, and X. Zhou, "Multi-objective path planning for uav in the urban environment based on cdnsga-ii," in *2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)*. IEEE, 2019, pp. 350–3505.

[3] N. Khial, N. Mhaisen, M. Mabrok, and A. Mohamed, "An online learning framework for uav search mission in adversarial environments," *Available at SSRN 4725375*.

[4] A. Soliman, A. Al-Ali, A. Mohamed, H. Gedawy, D. Izham, M. Bahri, A. Erbad, and M. Guizani, "Ai-based uav navigation framework with digital twin technology for mobile target visitation," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106318, 2023.

[5] H. Wang, Y. Peng, L. Liu, and J. Liang, "Study on target detection and tracking method of uav based on lidar," in *2021 Global Reliability and Prognostics and Health Management (PHM-Nanjing)*. IEEE, 2021, pp. 1–6.

[6] M. Theile, H. Bayerlein, R. Nai, D. Gesbert, and M. Caccamo, "Uav coverage path planning under varying power constraints using deep reinforcement learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1444–1449.

[7] H. X. Pham, H. M. La, D. Feil-Seifer, and L. Van Nguyen, "Reinforcement learning for autonomous uav navigation using function approximation," in *2018 IEEE international symposium on safety, security, and rescue robotics (SSRR)*. IEEE, 2018, pp. 1–6.

[8] S. Bhagat and P. Sujit, "Uav target tracking in urban environments using deep reinforcement learning," in *2020 International conference on unmanned aircraft systems (ICUAS)*. IEEE, 2020, pp. 694–701.

[9] H. Bayerlein, P. De Kerret, and D. Gesbert, "Trajectory optimization for autonomous flying base station via reinforcement learning," in *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2018, pp. 1–5.

[10] D. Ebrahimi, S. Sharafeddine, P.-H. Ho, and C. Assi, "Autonomous uav trajectory for localizing ground objects: A reinforcement learning approach," *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, pp. 1312–1324, 2020.

[11] C. Wu, B. Ju, Y. Wu, X. Lin, N. Xiong, G. Xu, H. Li, and X. Liang, "Uav autonomous target search based on deep reinforcement learning in complex disaster scene," *IEEE Access*, vol. 7, pp. 117 227–117 245, 2019.

[12] A. Tharwat, M. Elhoseny, A. E. Hassanien, T. Gabel, and A. Kumar, "Intelligent bézier curve-based path planning model using chaotic particle swarm optimization algorithm," *Cluster Computing*, vol. 22, pp. 4745–4766, 2019.

[13] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[14] Z. Chu, F. Wang, T. Lei, and C. Luo, "Path planning based on deep reinforcement learning for autonomous underwater vehicles under ocean current disturbance," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 108–120, 2022.

[15] J. Xue, X. Kong, B. Dong, and M. Xu, "Multi-agent path planning based on mpc and ddpg," *arXiv preprint arXiv:2102.13283*, 2021.

[16] H. Huang, Y. Yang, H. Wang, Z. Ding, H. Sari, and F. Adachi, "Deep reinforcement learning for uav navigation through massive mimo technique," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 1117–1121, 2019.

[17] O. Pérez-Gil, R. Barea, E. López-Guillén, L. M. Bergasa, C. Gómez-Huélamo, R. Gutiérrez, and A. Díaz-Díaz, "Deep reinforcement learning based control for autonomous vehicles in carla," *Multimedia Tools and Applications*, vol. 81, no. 3, pp. 3553–3576, 2022.

[18] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "Autonomous uav navigation: A ddpg-based deep reinforcement learning approach," in *2020 IEEE International Symposium on circuits and systems (ISCAS)*. IEEE, 2020, pp. 1–5.