frontiers | Frontiers in Robotics and AI

TYPE Original Research
PUBLISHED 03 February 2025
DOI 10.3389/frobt.2024.1527095

OPEN ACCESS

EDITED BY
Jun Ma,
Hong Kong University of Science and Technology, Hong Kong SAR, China

REVIEWED BY
Wenxin Wang,
National University of Singapore, Singapore
Huan Yu,
Zhejiang University, China
Pengqin Wang,
Hong Kong University of Science and Technology, Hong Kong SAR, China in collaboration with reviewer HY

*CORRESPONDENCE
Jan-Hendrik Ewers,
✉ j.ewers.1@research.gla.ac.uk

RECEIVED 12 November 2024
ACCEPTED 30 December 2024
PUBLISHED 03 February 2025

CITATION
Ewers J-H, Anderson D and Thomson D (2025) Deep reinforcement learning for time-critical wilderness search and rescue using drones.
Front. Robot. AI 11:1527095.
doi: 10.3389/frobt.2024.1527095

Deep reinforcement learning for time-critical wilderness search and rescue using drones

Jan-Hendrik Ewers*, David Anderson and Douglas Thomson

Autonomous Systems and Connectivity, University of Glasgow, Glasgow, United Kingdom

Traditional search and rescue methods in wilderness areas can be time-consuming and have limited coverage. Drones offer a faster and more flexible solution, but optimizing their search paths is crucial for effective operations. This paper proposes a novel algorithm using deep reinforcement learning to create efficient search paths for drones in wilderness environments. Our approach leverages *a priori* data about the search area and the missing person in the form of a probability distribution map. This allows the policy to learn optimal flight paths that maximize the probability of finding the missing person quickly. Experimental results show that our method achieves a significant improvement in search times compared to traditional coverage planning and search planning algorithms by over 160%, a difference that can mean life or death in real-world search operations Additionally, unlike previous work, our approach incorporates a continuous action space enabled by cubature, allowing for more nuanced flight patterns.

KEYWORDS

reinforcement learning, search planning, mission planning, autonomous systems, wilderness search and rescue, unmanned aerial vehicle, machine learning

1 Introduction

Wilderness Search and Rescue (WiSAR) operations in Scotland's vast and often treacherous wilderness pose significant challenges for emergency responders. To combat this, Police Scotland Air Support Unit (PSASU) and Scottish Mountain Rescue (SMR) regularly use helicopters to assist in search operations (Carrell, 2022). However, the deployment of helicopters can be slow, especially in the Scottish Isles where the PSASU's Eurocopter EC135 based in Glasgow can take multiple hours to arrive. Additionally, operating helicopters is extremely costly.

Drones, also known as unmanned aerial vehicles, offer a cost-effective and agile solution for aerial search. Both PSASU and SMR are placing small fleets of drones around Scotland for rapid deployment in a WiSAR scenario. These fleets will never replace the requirement for a helicopter due to the inherent lifting disparities between the two platforms but will ensure that the search can begin as soon as possible.

The current approach to flying drones for PSASU is the pilot-observer model where two personnel are required at a minimum per drone. In this setup, the observer is in charge of maintaining visual line of sight at all times whilst the pilot can fly the drone and inspect the live camera feed. Koester et al. (2004) identifies that a foot-based searcher has a higher detection rate when not in motion, similar to the behaviour exhibited by pilots (Ewers et al., 2023b). The cognitive load of being in-motion whilst searching is evidently a barrier for pilots and offloading one aspect of this may lead to efficiency gains.

To address this challenge and to also free up precious manpower, search planning algorithms are employed to optimize drone flight paths (Lin and Goodrich, 2009). While Deep Reinforcement Learning (DRL) has not been explored in this context, its success in other domains, such as video games and drone racing (Mnih et al., 2015; Kaufmann et al., 2023), suggests its potential for improving WiSAR search planning. The ability of DRL to generalize the problem and make decisions that maximise future gains based on extensive training allows it to provide unique solutions to the problem.

WiSAR search planning is an information rich task and the effective utilization of prior information - such as the place last seen, age, and fitness levels of the missing person - is critical. This information can be used to generate a Probability Distribution Map (PDM), which describes the probability of detecting the missing person at a given location and informs the search stage of the mission. Without this *a priori* information the search problem converts into either a coverage or exploration problem. The former assumes a uniform PDM whilst the later develops an understanding of the environment in real-time. There are a multitude of algorithms that can generate the PDM (Ewers et al., 2023a; Hashimoto et al., 2022; Šerić et al., 2021) and as such it can be assumed to be a known quantity during search planning.

The core contributions of this research to the field are thus as follows:

- We propose a novel application of DRL to the search planning problem which can outperform benchmark algorithms from the literature (Lin and Goodrich, 2009; Subramanian et al., 2020). This apporach leverages *a priori* information about the search space without limiting its field of view and thus reducing performance.
- We propose the use of a continuous PDM, as opposed to an image-based one (Lin and Goodrich, 2009; Subramanian et al., 2020), to prevent undesired noisy rewards (Guo and Wang, 2023; Fox et al., 2017). This further empowers the policy to use a continuous action space which greatly increases the degrees of freedom over the benchmark algorithms from the literature.
- A framework for calculating the accumulated probability over the search path through cubature (Ewers et al., 2024) is introduced. A different formulation for this calculation from the literature is required due to the use of the continuous PDM and action space.

Related work is discussed in Section 2, and the methodology is presented in Section 3. Results are shown in Section 4, and a conclusion is drawn in Section 5.

2 Related work

Coverage planning algorithms have been around for decades (Galceran and Carreras, 2013) in various forms with the most well-known, and intuitive, being the parallel swaths (also known as lawnmower or zig-zag) pattern. This guarantees complete coverage of an entire area given enough time. However, for WiSAR applications, reducing the time to find is substantially more important whilst also dealing with endurance constrained systems like drones.

Lin and Goodrich (2009) approach the search planning problem by using a gradient descent algorithm in the form of Local Hill Climbing (LHC) that can advance into any of the surrounding eight cells. However, LHC alone is not sufficient because Waharte and Trigoni (2010) found that this class of algorithm does not perform well due to their propensity in getting stuck around local maxima. For this reason Lin and Goodrich (2009) introduces the notion of global warming to break out of local maxima. This raises the zero probability floor sequentially a number of times, storing the paths and then reassessing them given the original PDM. Through this, and a convolution-based tie-breaking scheme, LHC_GW_CONV (local hill climb, global warming, convolution) is shown to have very favourable results. However, only the adjacent areas are considered at every time step.

In order to consider the area as a whole, sampling-based optimisation approaches have been applied to the problem. Morin et al. (2023) uses ant colony optimisation with a discrete PDM and Ewers et al. (2023b) uses both genetic algorithm and particle swarm optimisation with a pseudo-continuous PDM. However, due to the nature of sampling-based optimisation problems, they are prone to long computation times to converge on a solution.

A core problem with the previously mentioned algorithms is the inability to consider the PDM as a whole when making decisions. Being able to prioritise long-term goals over short-term gains is a key feature of DRL.

DRL is being used extensively for mission planning such as by Yuksek et al. (2021) who used proximal policy optimisation to create a trajectory for two drones to avoid no-fly-zones whilst tracking towards the mission objective. This approach has defined start and target locations, however the uses of no-fly-zones with constant radius is analogous to an inverted PDM. Peake et al. (2020) uses Recurrent-DDQN for target search in conjunction with A2C for region exploration with a single drone to find missing people. This method does not use any *a priori* information but rather explores the area in real time. This shows that DRL is a suitable approach to the search-over-PDM problem that a WiSAR mission requires.

As highlighted in Section 1, search and exploration planning are very different problems. Exploration planning has seen many different planning algorithms such as in work by Zhang et al. (2024) which uses a partially observable markov decision process and environment discretisation handle exploration and search in near real time. Similarly, Talha et al. (2022) and Peake et al. (2020) use DRL with environment discretisation to explore the environment whilst searching. Ebrahimi et al. (2021) also uses DRL but localizes missing people using radio signal strength indexes without prior knowledge of the searchable domain. These algorithms do not have any *a priori* knowledge available during the planning stage other than what is discovered in real-time; a different problem.

A core aspect of DRL is having a fully observable environment such that the policy can infer why the action resulted in the reward. Thus, being able to represent the PDM effectively is a primary goal. Whilst images can be used as inputs for DRL, as done by Mnih et al. (2015) to play Atari 2600 computer games, the typically large dimension can be prohibitive. However, since PDM generation algorithms most commonly create discrete maps (Šerić et al.,

TABLE 1 Simulation parameters used for this study.

Parameter	Value	Units
N_{gaussian}	4	
$\boldsymbol{\sigma}_i$	$diag(500, 500)$	
$x_{\text{min}}, y_{\text{max}}$	0	m
$x_{\text{min}}, y_{\text{max}}$	150	m
λ	8	m
R_{buffer}	2.5	m
N_{waypoint}	64	
ϵ	0.1	
w_{oob}	1.0	
w_r	0.5	
w_0	0.5	

TABLE 2 SAC hyperparameters used for this study from empirical testing. Other variables were kept at the default values from Raffin et al. (2021) `v2.1.0`.

Hyperparameter	Value
Learning rate	10^{-6}
Optimizer	AdamKingma and Ba (2017)
Batch size	1024
Learning starts	8192
Buffer size	5×10^{6}
Training frequency	10
Gradient steps	50
τ	10^{-4}

2021; Hashimoto et al., 2022), finding a different representation is required. To go from a discrete to continuous PDM, Lin and Goodrich (2014) uses a gaussian mixture model to represent the PDM as a sum of bivariate Gaussians. This can be easily used to numerically represent the numerous bivariate Gaussian parameters in an array which is a suitable format for a DRL observation.

There are many DRL algorithms to chose from with Proximal Policy Optimisation (PPO) (Schulman et al., 2017) and Soft Actor-Critic (SAC) (Haarnoja et al., 2018) being some of the most prevalent in the literature (Yuksek et al., 2021; Mock and Muknahallipatna, 2023; Xu et al., 2019). Mock and Muknahallipatna (2023) found that PPO performed well for low dimension observation spaces, whilst SAC performed much better for larger ones. The need for a large observation space comes from the fact that the policy would need to have a sense of memory regarding where it had been to encounter *unseen* probability to satisfy the markov decision process that underpins DRL. Mock and Muknahallipatna (2023) found that a recurrent architecture was comparable to including previous states in the observation (also known as frame stacking). This shows that frame-stacking with SAC is a suitable DRL architecture for the current problem.

3 Methods

All parameters used in this study can be found in Tables 1, 2.

3.1 Modelling

3.1.1 Environment

The low level tasks of control (Tedrake, 2023; Fresk and Nikolakopoulos, 2013; Wang et al., 2023), trajectory generation (Yu et al., 2023), and obstacle avoidance (Richter et al., 2016; Levine et al., 2010; Swinton et al., 2024) can be assumed to be of a high enough standard as to achieve perfect waypoint mission execution performance. The drone within the environment is therefore modelled as a simple heading control model with a constant step size λ. Thus, the position vector $\mathbf{x} \in \mathbb{R}^2$ is updated via.

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \lambda \begin{bmatrix} \cos u_t \\ \sin u_t \end{bmatrix}$$
$$u_t = \pi\left(a_t + 1\right) \tag{1}$$

where $a_t \in [-1, 1]$ is the policy action at time-step t.

From Equation 1 it is clear that the state $\mathbf{x}_{t=T}$ is dependent on the states from $t = 0$ to $t = T$ making this model suitable for formulating the drone's motion as a markov decision process (MDP). We define the tuple (S, A, P, R) where.

- $s \in S = \mathbb{R}^2$ is the finite set of states, representing the possible positions of the drone.
- $a \in [-1, 1]$ is the action space.
- $P{:}S \times A \times S \rightarrow [0, \infty]$ is the unknown transition probability function.
- $A{:}S \times A \rightarrow \mathbb{R}$ is the reward function.

We further define the reward function in Section 3.1.3 and outline the training of the optimal policy $\pi^*(s|a)$ in Section 3.2.

3.1.2 Probability distribution map

In a real WiSAR scenario, algorithms such as Ewers et al. (2023a), Hashimoto et al. (2022), and Šerić et al. (2021) can be employed to generate the PDM given the search mission profile - last place seen, terrain, profile of the lost person, and more. This data is not publicly available in any meaningful quantity and is thus not usable in this scenario. Therefore, as is common within the literature, the PDM is randomly generated for training and evaluation.

The PDM is modelled as a sum of N_{gaussian} bivariate Gaussians (Yao et al., 2019) such that a point on the ground at coordinate $\mathbf{x} \in \mathbb{R}^2$

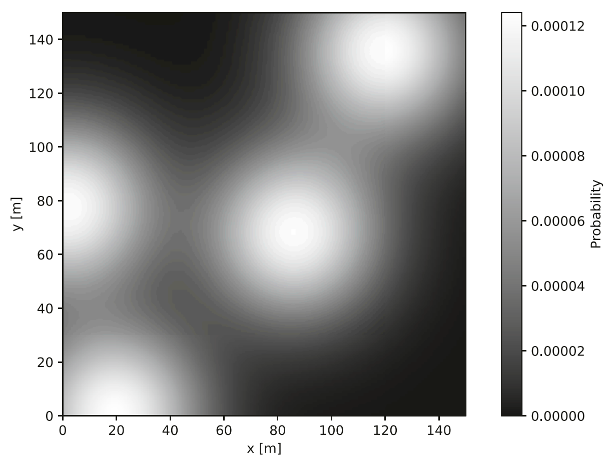

FIGURE 1
An example multi-modal bivariate Gaussian PDM.

has a probability of containing the missing person.

$$p(\mathbf{x}) = \frac{1}{N_{\text{gaussian}}} \sum_{i=0}^{N_{\text{gaussian}}} \frac{\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right]}{\sqrt{4\pi^2 \det \boldsymbol{\sigma}_i}}$$
$$\forall i \in [0,G], \boldsymbol{\mu}_i \sim U\left([x_{\min},x_{\max}],[y_{\min},y_{\max}]\right) \qquad (2)$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$ are the mean location and covariance matrix of the ith bivariate Gaussian respectively. If the bounding area were infinite, that is $x_{\min} = y_{\min} = -\infty$ and $x_{\max} = y_{\max} = \infty$, then $\sum p(\mathbf{x}) = 1$. However, as can be seen from Figure 1, the area enclosed by the rectangular bounds contain less than this. Section 3.1.3 further discusses how this is handled such that the available probability is normalized.

3.1.3 Reward

As the agent moves a constant distance sm every step, it is assumed that the camera follows this path continuously at a fixed height whilst pointing straight down at all times. Therefore, to represent the *seen area* for a given path at time-step t, the path is buffered by R_{buffer}m to give the polygon h_t. All probability from the PDM enclosed within h_t is then *seen* and denoted by p_t. This value, the seen probability, is calculated through

$$I(H) = \int_H p(\mathbf{x})\, dH$$

with $H = h_t$. $p(\mathbf{x})$ is from Equation 2. Thusly,

$$p_t = I(h_t)$$

The integral is calculated using a cubature integration scheme (Ewers et al., 2024) with constrained Delaunay triangulation (Chew, 1987) to subdivide H into triangles as seen in Figure 2B.

Other than allowing easy calculation of the accumulated probability, the buffering of the path prevents revisiting of an area contributing the same probability multiple times. This can be seen at the cross-over point $(2.5, 2.5)$m in Figure 2A.

In order to correlate action to reward, only the additional probability that has been accumulated

$$\Delta p_t = p_t - p_{t-1} \qquad (3)$$

is used. To normalize this value, the scaling constant k is introduced. This scales Δp_t by the ratio of the area of an isolated step dm to the area of the total search area $a_{\text{area}}\text{m}^2$. This is defined as

$$k = \frac{a_{\text{area}}}{R_{\text{buffer}}\left(\pi R_{\text{buffer}} + 2\lambda\right)} \qquad (4)$$

with further spatial definitions from Figure 3.

As highlighted in Section 3.1.2, the enclosed probability by the bounds is not equal to 1. To handle this, Δp_t is scaled by the total available probability within the search area $p_A = I(A)$. Combining p_A with Equations 3, 4, gives the reward

$$r = \frac{k}{p_A} \Delta p_t$$

The enclosed probability can then be used to calculate the probability efficiency at time-step t with

$$e_{p,t} = \frac{p_t}{p_A} \qquad (5)$$

and $e_{p,t} \leq 1$.

Finally, reward shaping is used to discourage future out-of-bounds actions and to penalize visiting areas of low probability or revisiting previously seen sections. The latter is easily handled by the buffering of the path as seen in Figure 2A, where the areas highlighted in red will contribute no value to the reward resulting in a penalty of $-w_{oob}$. The augmented reward r' is then defined as

$$r' = \begin{cases} -w_{oob}, & \mathbf{x}_t \notin [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \\ w_r r, & \Delta p_t > \epsilon \\ -w_0, & \text{else} \end{cases}$$

3.2 Training algorithm

SAC is a DLR algorithm particularly effective for continuous control tasks and it addresses the exploration-exploitation dilemma by simultaneously maximizing expected reward and entropy. The interaction of the policy with the environment can be seen in Figure 4. Entropy, a measure of uncertainty, encourages the agent to explore the environment, preventing it from getting stuck in suboptimal solutions.

SAC employs an actor-critic architecture consisting of the following key components:

- Policy Network (Actor): Denoted as $\pi_\phi(a|s)$, it represents the current policy parameterized by ϕ. SAC is a stochastic algorithm and thus actions are sampled from the policy through $a_t \sim \pi_\phi(a_t|s_t)$.
- Q-function Networks (Critics): SAC uses two soft Q-networks, $Q_{\theta_1}(s,a)$ and $Q_{\theta_2}(s,a)$, parameterized by θ_1 and θ_2 respectively. These estimate the expected cumulative reward for taking action a in state s and following the policy thereafter.

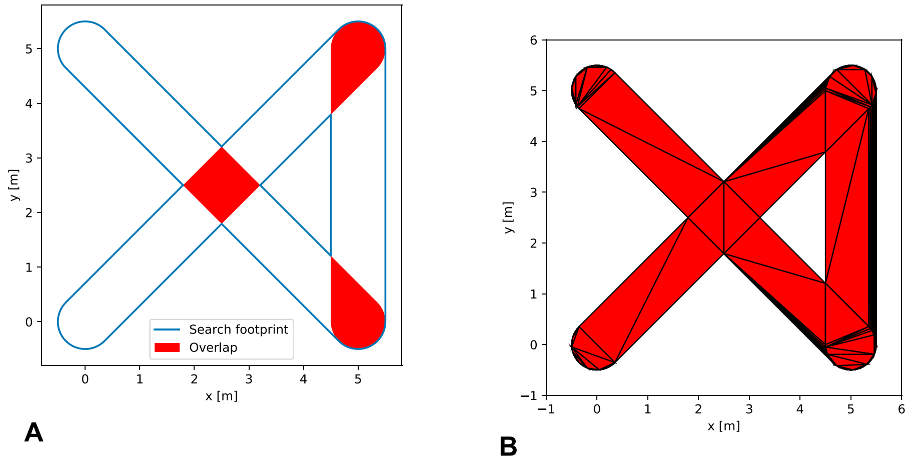

FIGURE 2
Visualizations of concepts related to the buffered polygon representation of the seen area. **(A)** An example of how the buffered path polygon *h* automatically deals with re-seen areas. Note that the highlighted areas are just for demonstration and are not a part of the algorithm. **(B)** The polygon from **(A)** triangulated using the Delauney constrained triangulation.

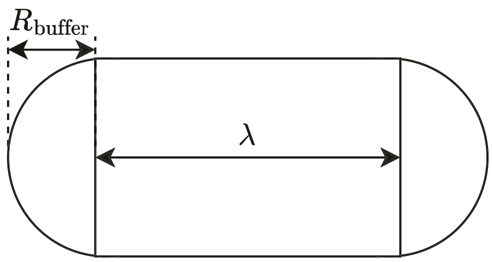

FIGURE 3
The anatomy of the area calculation of an isolated step of length λ with buffer R_{buffer} as used in Equation 4.

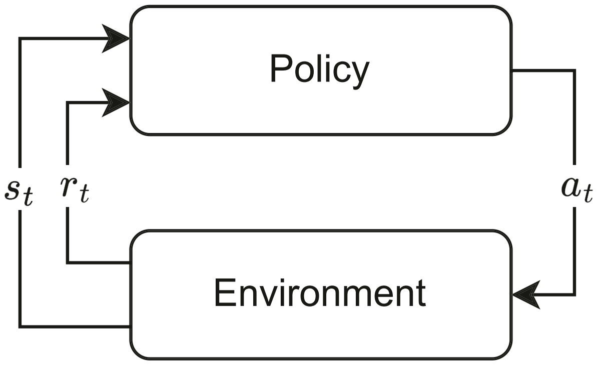

FIGURE 4
Top-level representation of a typical reinforcement learning data flow. The agent is also commonly referred to as the policy.

- Value Network: While not explicitly maintained, the soft state-value function $V(s)$ is implicitly defined as:

TABLE 3 Definition of the five state observations.

Sub-state	Symbol	Definition
Path	s_{path}	$\left(\mathbf{x} \,\|\, \mathbf{0}^{2\times N_{\text{waypoints}}-t}\right)^T$
PDM	s_{PDM}	$[\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0, \ldots, \boldsymbol{\mu}_G, \boldsymbol{\sigma}_G]^T$
Position	s_{pos}	$\mathbf{x}_t$
Out-of-bounds	s_{oob}	$\mathbf{x}_t \in [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$
Number of steps	s_{steps}	t

$$V(s_t) = \mathbb{E}_{a_t \sim \pi_\phi}\left[Q(s_t, a_t) - \alpha \log \pi_\phi(a_t|s_t)\right]$$

The use of two soft Q-networks helps to reduce positive bias in the policy improvement step, a common issue in value-based methods. The soft Q-networks can be trained to minimize the soft Bellman residual:

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim D}\left[\frac{1}{2}\left(Q_\theta(s_t, a_t) - \left(r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}\sim p}\left[V_{\bar{\theta}}(s_{t+1})\right]\right)\right)^2\right]$$

SAC incorporates an automatic entropy tuning mechanism (Haarnoja et al., 2019) to adjust the temperature parameter α during training. This allows the algorithm to adapt the degree of exploration based on the policy's performance. α is learnt by minimizing the following loss before the target network is updated with:

$$J(\alpha_t) = \mathbb{E}_{a_t \sim \pi_t}\left[-\alpha_t \log \pi_t(a_t|s_t; \alpha_t) - \alpha_t H(\pi_\theta(\cdot|s_t))\right]$$

where $H(\pi_\theta(\cdot|s_t))$ is the entropy of the policy π_θ given the state s_t.

By automatically tuning α, SAC can maintain an appropriate balance between exploration and exploitation throughout the learning process, adapting to the complexity of the task and the stage of learning.

3.2.1 Policy network

The core of the policy network $\pi_{\theta,\text{core}}$ consists of a fully connected network with N_{layers} layers, each with a width of N_{width}. The observation definitions are given in Table 3 resulting in $2N_{\text{waypoints}}+6G+4$ observation inputs. The policy π_ϕ is constructed to handle multiple input observation spaces and is defined in Figure 5. The inner workings of the policy are defined in Algorithm 1.

4 Results

4.1 Experimental setup

In order to effectively benchmark the proposed algorithms, two additional baselines are implemented; lawnmower (Galceran and Carreras, 2013) and LHC_GW_CONV (Lin and Goodrich, 2009). These were chosen due to the former being ubiquitous for coverage planning, and the latter being a optimisation-based implementation that struggles to fully explore the PDM. To ensure compatibility in the comparison to the proposed algorithm, the parallel lines for lawnmower are offset by the step size λ and the grid dimensions for LHC_GW_CONV are $(x_{\max}-x_{\min}, y_{\max}-y_{\min})/\lambda$. The maximum number of waypoints N_{waypoint} are converted to a maximum distance $D_{\max}=\lambda N_{\text{waypoint}}$ and the generated paths are truncated at this point.

The results for the algorithm implemented in this research, titled *SAC-FS-CNN* from here on in, is the cumulation of three separate training runs with random starting seeds. This aligns with the best practices outlined by Agarwal et al. (2022) to ensure robust analysis for DRL results. Each model was trained for a minimum of 21 days (5×10^8 global steps) with 32 workers on a local `Ubuntu 22.04` machine with a `AMD Ryzen 9 5950X` CPU, a `NVIDIA RTX A6000` GPU, and 64GB of RAM.

One evaluation of an algorithm involves generating the random PDM, then creating the resultant search path. This is labelled one run. Each algorithm was evaluated at least 5×10^3 times and this generated data is base of the following analysis.

4.2 Probability over distance (POD)

Maximising the probability efficiency (Equation 5) at all times is critical. This directly correlates to increasing the chances of finding a missing person in a shorter time. It is important for the POD of SAC-FS-CNN to out-perform the benchmark algorithms at all times. If this is not the case, the search algorithm selection becomes dependent on the endurance and mission. However, if the POD is better at all times then one algorithm will be superior no matter the application. To calculate the POD, the probability efficiency is evaluated at

$$d=\frac{\left(N_{\text{steps}}-i\right)D}{N_{\text{steps}}}\forall i\in\left\{N_{\text{steps}},N_{\text{steps}}-1,\ldots,1,0\right\} \tag{6}$$

with $N_{\text{steps}}=50$.

From Figure 6A it is clear that SAC-FS-CNN sufficiently outperforms the benchmark algorithms at all distances. This is further highlighted by the $e_{p,D}$ for SAC-FS-CNN at 238% of that of lawnmower, and 158% for LHC_GW_CONV from Table 4. This is corroborated by the median $e_{p,D}$ values in Figure 6C. Notably, however, LHC_GW_CONV has a substantial amount of high $e_{p,D}$ outliers.

Likewise, the performance profile from Figure 6B follows the trend. It can be seen that SAC-FS-CNN has close to 100% of runs

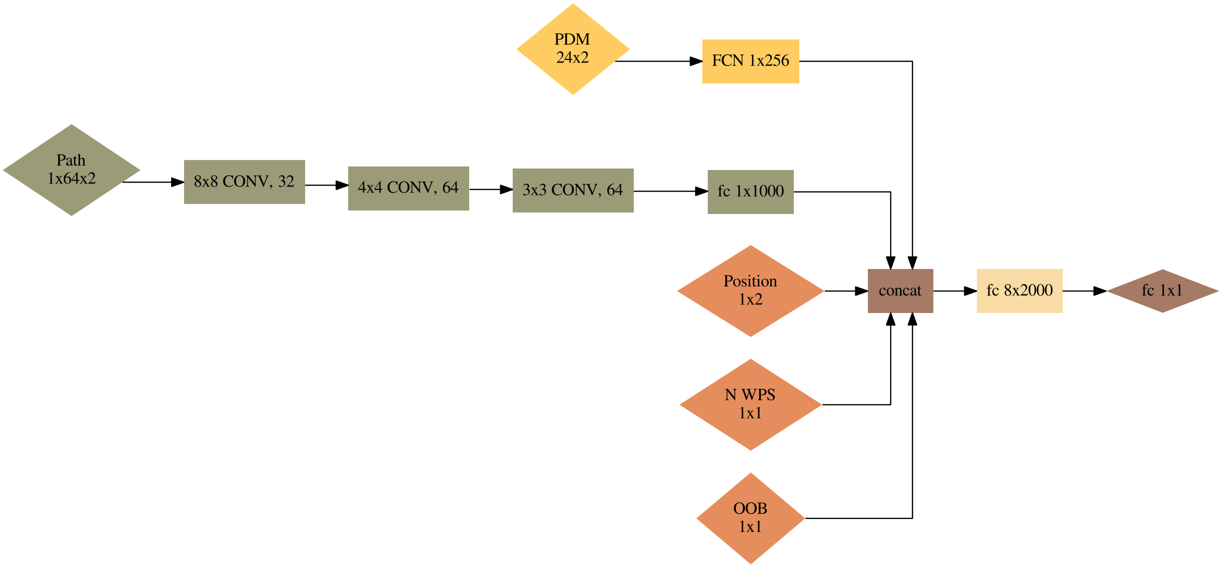

FIGURE 5
Policy network π_ϕ. $\pi_{\theta,\text{path}}$ uses a 2D CNN architecture from Mnih et al. (2015).

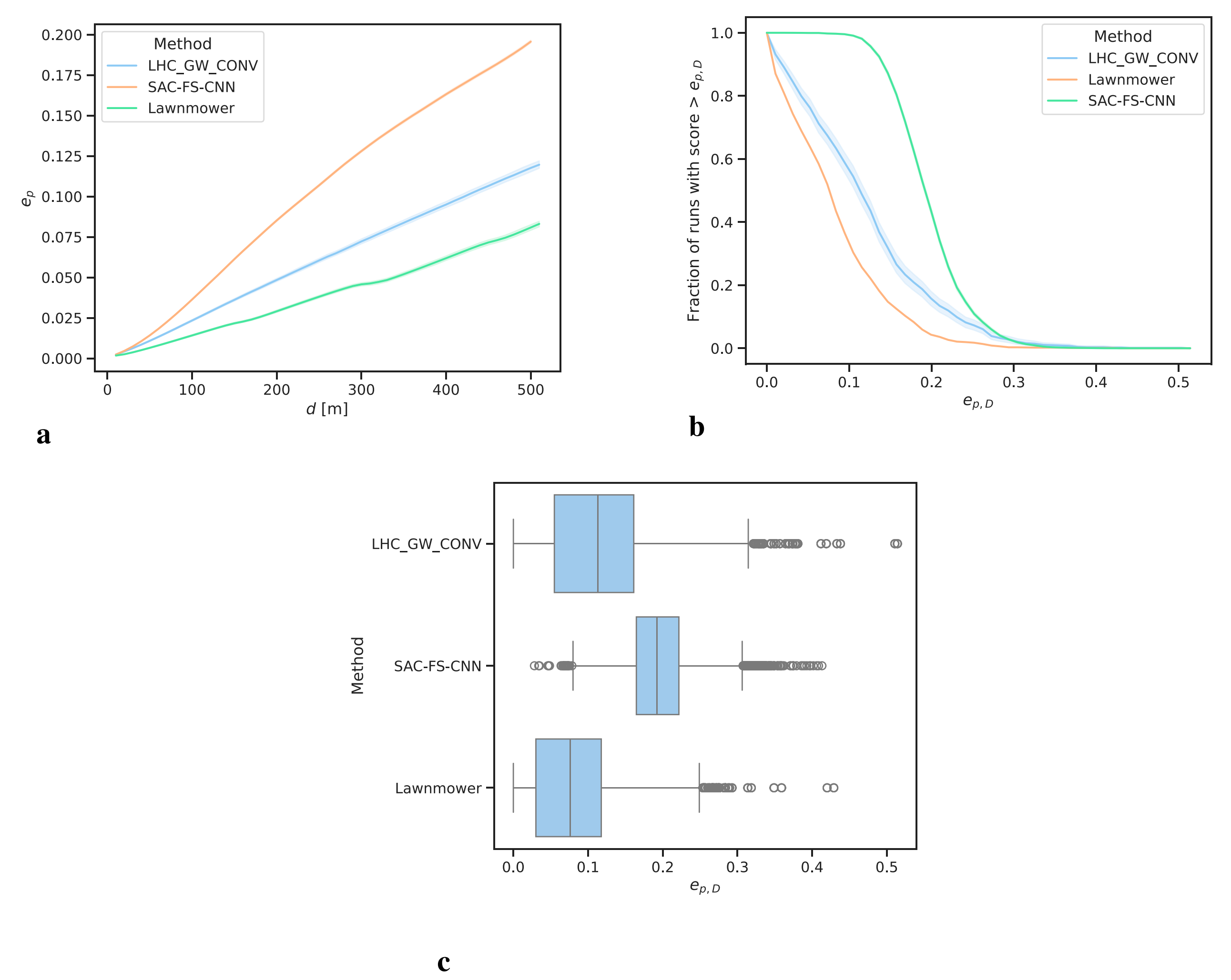

FIGURE 6
Probability efficiency analysis over all runs. The shaded regions in **(A, B)** show the 95% confidence interval. **(A)** Mean e_p with d showing SAC-FS-CNN outperforming the benchmark algorithms at all distances. **(B)** Performance profile of $e_{p,D}$. SAC-FS-CNN has a close to 100% of runs with $e_{p,D} > 0.1$ compared to the $< 50\%$ for the other algorithms. **(C)** Median $e_{p,D}$ for all runs. SAC-FS-CNN has a higher median than the other algorithms. LHC_GW_CONV registers large outliers.

TABLE 4 Mean POD with the standard deviation as error.

Method	p_D	$e_{p,D}$	N
LHC_GW_CONV	0.09 ± 0.06	0.12 ± 0.08	9.3×10^3
Lawnmower	0.06 ± 0.04	0.08 ± 0.06	10×10^3
SAC-FS-CNN	$\mathbf{0.15 \pm 0.04}$	$\mathbf{0.19 \pm 0.04}$	9.8×10^3

with $e_{p,D} > 0.1$ and 50% at approximately $e_{p,D} > 0.2$. This aligns with results from Figure 6C and Table 4.

It is of note that LHC_GW_CONV has the largest range of values going from 0.0 to 0.5 whilst SAC-FS-CNN only goes from 0.02 to 0.41. This shows that LHC_GW_CONV can perform very well given the right PDM or poorly given the wrong one. The DLR approach of SAC-FS-CNN, on the other hand, does not suffer from this due to its ability to find a general solution to the problem.

4.3 Distance to find (DTF) and percentage found (PF)

Whilst POD shows the theoretical effectiveness of an algorithm, the intended use-case is finding a missing person whilst searching within a bounded area. The mission statement is reducing the time it takes to find the potentially vulnerable person to save lives.

To quantify this requirement, we introduce DTF and PF. The former gives a clear answer on the capabilities on the various algorithms, whilst the latter should align with the POD results from Table 4 for validation.

Firstly, Gumbel-Softmax (Li et al., 2021) is used to sample N_{samples} positions from the PDM to give the set $\chi \in \mathbb{R}^{2\times N_{\text{samples}}}$ containing all samples. The path is then traversed in incremental steps using Equation 6 with $N_{\text{step}} = 10^4$. At each step, a euclidean distance check is done from the current position $\mathbf{x}$ to each entry in χ with any points within R_{buffer} being marked as seen. The updated

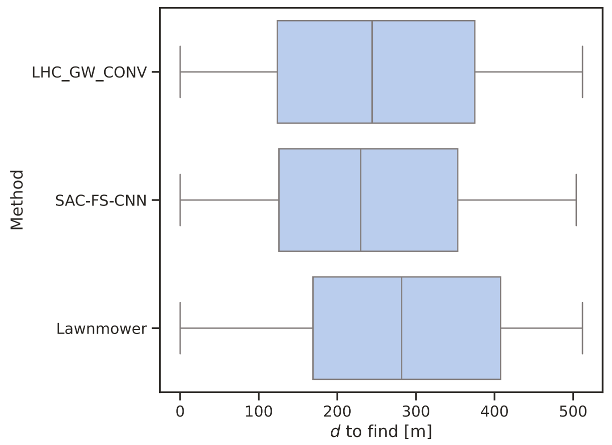

FIGURE 7
Median *d* to find. Similarly, the whisker markers across the algorithms shows the broad distribution of results from this experiment. SAC-FS-CNN ultimately has the lowest median score.

TABLE 5 DTF with the standard deviation as error.

Method	PF [%]	Mean DTF [m]	N
LHC_GW_CONV	11.86 ± 0.32	249.37 ± 145.70	5×10^6
Lawnmower	7.77 ± 0.27	282.63 ± 146.62	5×10^6
SAC-FS-CNN	**19.00 ± 0.39**	**239.61 ± 138.16**	10×10^6

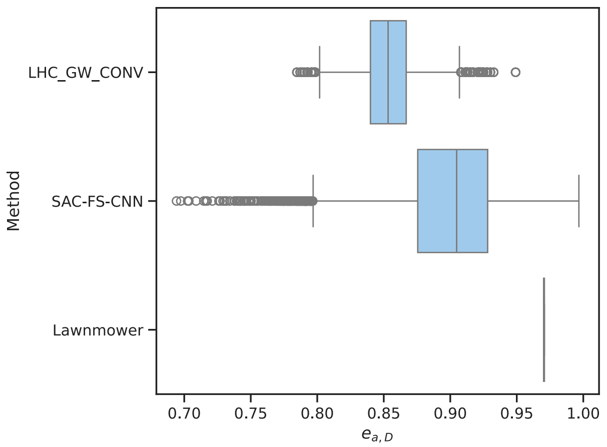

FIGURE 8
Area efficiency of the three algorithms. Lawnmower has many straight segments which result in a high score of 0.97 and no variance because it does not change. SAC-FS-CNN on the other hand outperforms the other environment-dependent algorithm LHC_GW_CONV in this metric.

```
Input: Observations defined in Table 3
Output: Action a
Extract PDM latent space $z_{\mathrm{PDM}} \leftarrow \pi_{\theta,\mathrm{PDM}}(s_{\mathrm{PDM}})$;
Extract path history latent space
$z_{\mathrm{path}} \leftarrow \pi_{\theta,\mathrm{path}}(s_{\mathrm{path}})$;
Concatenate remaining observations and latent
spaces $z \leftarrow z_{\mathrm{PDM}} \cup z_{\mathrm{path}} \cup [s_{\mathrm{oob}}, s_{\mathrm{pos}}, s_{\mathrm{steps}}]$;
Sample from policy to get action $a \sim \pi_{\theta,\mathrm{core}}(z)$
```

Algorithm 1. Policy Network π_ϕ Input Process.

set of positions to search for in the next step is then

$$\chi' = \left\{\chi_i \in \chi : \|\mathbf{x} - \chi_i\| > R_{buffer}\right\}$$

From Figure 7, it is clear to see that SAC-FS-CNN outperforms the benchmark algorithms with a lower median DTF as well as a lower inter-quartile range. Table 5 shows that the mean DTF is 15.22% lower than lawnmower, and 4.07% lower than LHC_GW_CONV. This is in line with expectations from the results in Section 4.2. Likewise, the PF values closely match to the $e_{p,D}$ values from Table 4 showing that this test correlates to the theory.

Whilst SAC-FS-CNN outperforms the benchmarks in the median DTF, it is of note that the variances of the three algorithms are almost identical as shown by the whiskers in Figure 7. This is due to the manner in which the positions are sampled from the random PDM making it likely for there to be a very small subset of positions near the start and end of the path. It is evident that this is the cause because the variances of the three algorithms range from approximately 0 to 512 which is the full simulation distance.

4.4 Area efficiency

A path with corners has overlapping regions when considering the buffered path which is evident from Figure 2A. The most efficient path in this formulation is thus a straight line such that the area efficiency is $e_{a,D} = 1$. Using Equation 4, this value is calculated with

$$e_{a,D} = \frac{a_{\mathrm{buffer,D}}}{R_{buffer}\left(\pi R_{buffer} + 2D\lambda\right)}$$

where $a_{\mathrm{buffer,D}}$ is the total area of the buffered path, and D is the number of waypoints in the path.

The aggregated metrics can be seen in Figure 8 which shows a significant difference in area efficiency between the three methods. The lawnmower method consistently achieves the highest area efficiency values, with a median value of 0.97. This suggests that lawnmower generates paths with minimal overlapping regions within the buffer, resulting in efficient utilization of the search buffer. However, lawnmower has no variance in area efficiency as it is a coverage planning algorithm and as such always generates the same path. In contrast, the SAC-FS-CNN method demonstrates lower area efficiency, with a median value around 0.90. This indicates that SAC-FS-CNN paths tend to have more overlapping areas within the buffer, leading to suboptimal utilization. LHC_GW_CONV method exhibits the lowest area efficiency, with a median value of 0.86 due its inability to make trade-offs now for future gains due to its local hill climbing formulation. This, however, is not the case for LHC_GW_CONV as an infinite-horizon discounted reward is at the core of the

SAC algorithm meaning that the current action is taken in order to maximise the future rewards.

5 Conclusion

Our research proposed SAC-FS-CNN for search planning in WiSAR operations, leveraging *a priori* information. This was identified as a solution to the challenge of maximizing accumulated probability over a given area due to the powerful capabilities of machine learning to identify patterns and make generalizations in complex tasks.

The results indicate that SAC-FS-CNN can outperform benchmark algorithms in the probability efficiency by up to 250% for lawnmower, and 166% for LHC_GW_CONV. A similar trend is identified when comparing mean DTF with DRL outperforming the aforementioned algorithms by 15.22% and 4.07% respectively. The critical result, however, was that SAC-FS-CNN found 160% more simulated missing people than LHC_GW_CONV. This translates to a substantial advantage in locating missing individuals, potentially saving countless lives during WiSAR operations.

While SAC-FS-CNN exhibits lower area efficiency compared to lawnmower, this trade-off is justified by its superior performance in terms of POD and DTF. Future work could focus on improving area efficiency while maintaining its strong performance in these critical metrics.

The integration of DRL into WiSAR mission planning holds great potential for the future of search, offering a powerful tool with potential to significantly increase the success rate of WiSAR efforts.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

J-HE: Conceptualization, Data curation, Formal Analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualization, Writing–original draft, Writing–review and editing. DA: Conceptualization, Funding acquisition, Supervision, Writing–review and editing. DT: Supervision, Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Engineering and Physical Sciences Research Council, Grant/Award Number: EP/T517896/1-312561-05.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that no Generative AI was used in the creation of this manuscript.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frobt.2024.1527095/full#supplementary-material

References

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., and Bellemare, M. G. (2022). Deep reinforcement learning at the edge of the statistical precipice. Available at: http://arxiv.org/abs/2108.13264.

Carrell, S. (2022). Flying to the rescue: Scottish mountain teams are turning to drones. *Guard.* Available at: https://www.jmlr.org/papers/v22/20-1364.html.

Chew, L. P. (1987). Constrained delaunay triangulations. *Algorithmica* 4, 97–108. doi:10.1007/BF01553881

Ebrahimi, D., Sharafeddine, S., Ho, P.-H., and Assi, C. (2021). Autonomous uav trajectory for localizing ground objects: a reinforcement learning approach. *IEEE Trans. Mob. Comput.* 20, 1312–1324. doi:10.1109/TMC.2020.2966989

Ewers, J.-H., Anderson, D., and Thomson, D. (2023a). "GIS data driven probability map generation for search and rescue using agents," in *IFAC world congress 2023*, 1466–1471. doi:10.1016/j.ifacol.2023.10.1834

Ewers, J.-H., Anderson, D., and Thomson, D. (2023b). Optimal path planning using psychological profiling in drone-assisted missing person search. *Adv. Control Appl.* 5, e167. doi:10.1002/adc2.167

Ewers, J.-H., Sarah, S., Anderson, D., Euan, M., and Thomson, D. (2024). Enhancing reinforcement learning in sensor fusion: a comparative analysis of cubature and sampling-based integration methods for rover search planning *arXiv*, 7825–7830. doi:10.1109/iros58592.2024.10801978

Fox, R., Pakman, A., and Tishby, N. (2017). Taming the noise in reinforcement learning via soft updates *arXiv*. doi:10.48550/arXiv.1512.08562

Fresk, E., and Nikolakopoulos, G. (2013). "Full quaternion based attitude control for a quadrotor," in 2013 European Control Conference (ECC) (Zurich: IEEE), 3864–3869.

Galceran, E., and Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics Aut. Syst.* 61, 1258–1276. doi:10.1016/j.robot.2013.09.004

Guo, Y., and Wang, W. (2023). A robust adaptive linear regression method for severe noise. *Knowl. Inf. Syst.* 65, 4613–4653. doi:10.1007/s10115-023-01924-4

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor *arXiv*. doi:10.48550/arXiv.1801.01290

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., et al. (2019). Soft actor-critic algorithms and applications. Available at: http://arxiv.org/abs/1812.05905.

Hashimoto, A., Heintzman, L., Koester, R., and Abaid, N. (2022). An agent-based model reveals lost person behavior based on data from wilderness search and rescue. *Sci. Rep.* 12, 5873. doi:10.1038/s41598-022-09502-4

Kaufmann, E., Bauersfeld, L., Loquercio, A., Müller, M., Koltun, V., and Scaramuzza, D. (2023). Champion-level drone racing using deep reinforcement learning. *Nature* 620, 982–987. doi:10.1038/s41586-023-06419-4

Kingma, D. P., and Ba, J. (2017). Adam: a method for stochastic optimization *arXiv*. doi:10.48550/arXiv.1412.6980

Koester, R., Cooper, D. C., Frost, J. R., and Robe, R. Q. (2004). Sweep width estimation for ground search and rescue. Available at: https://www.semanticscholar.org/paper/Sweep-Width-Estimation-for-Ground-Search-and-Rescue-Koester-Cooper/10b35a96e9f34ae0f69a326bd33c6ba0db9fa172.

Levine, D., Luders, B., and How, J. (2010). "Information-rich path planning with general constraints using rapidly-exploring random trees," in *AIAA Infotech@Aerospace 2010* (Atlanta, Georgia: American Institute of Aeronautics and Astronautics).

Li, J., Chen, T., Shi, R., Lou, Y., Li, Y.-L., and Lu, C. (2021). Localization with sampling-argmax *arXiv*. doi:10.48550/arXiv.2110.08825

Lin, L., and Goodrich, M. A. (2009). UAV intelligent path planning for wilderness search and rescue. *2009 IEEE/RSJ Int. Conf. Intelligent Robots Syst.*, 709–714. doi:10.1109/IROS.2009.5354455

Lin, L., and Goodrich, M. A. (2014). Hierarchical heuristic search using a Gaussian mixture model for UAV coverage planning. *IEEE Trans. Cybern.* 44, 2532–2544. doi:10.1109/TCYB.2014.2309898

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. *Nature* 518, 529–533. doi:10.1038/nature14236

Mock, J. W., and Muknahallipatna, S. S. (2023). A comparison of PPO, TD3 and SAC reinforcement algorithms for quadruped walking gait generation. *J. Intelligent Learn. Syst. Appl.* 15, 36–56. doi:10.4236/jilsa.2023.151003

Morin, M., Abi-Zeid, I., and Quimper, C.-G. (2023). Ant colony optimization for path planning in search and rescue operations. *Eur. J. Operational Res.* 305, 53–63. doi:10.1016/j.ejor.2022.06.019

Peake, A., McCalmon, J., Zhang, Y., Raiford, B., and Alqahtani, S. (2020). Wilderness search and rescue missions using deep reinforcement learning. *2020 IEEE Int. Symposium Saf. Secur. Rescue Robotics (SSRR)*, 102–107. doi:10.1109/SSRR50563.2020.9292613

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N. (2021). Stable-Baselines3: reliable reinforcement learning implementations. *J. Mach. Learn. Res.* doi:10.5281/zenodo.11077334

Richter, C., Bry, A., and Roy, N. (2016). "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics research*. Editors M. Inaba, and P. Corke (Cham: Springer International Publishing), 114, 649–666. doi:10.1007/978-3-319-28872-7_37

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms *arXiv*. doi:10.48550/arXiv.1707.06347

Šerić, L., Pinjušić, T., Topić, K., and Blažević, T. (2021). Lost person search area prediction based on regression and transfer learning models. *ISPRS Int. J. Geo-Information* 10, 80. doi:10.3390/ijgi10020080

Subramanian, A., Alimo, R., Bewley, T., and Gill, P. (2020). "A probabilistic path planning framework for optimizing feasible trajectories of autonomous search vehicles leveraging the projected-search reduced hessian method," in *AIAA scitech 2020 forum* (Orlando, FL: American Institute of Aeronautics and Astronautics).

Swinton, S., Ewers, J.-H., McGookin, E., Anderson, D., and Thomson, D. (2024). A novel methodology for autonomous planetary exploration using multi-robot teams. Available at: https://arxiv.org/abs/2405.12790.

Talha, M., Hussein, A., and Hossny, M. (2022). "Autonomous UAV navigation in wilderness search-and-rescue operations using deep reinforcement learning," in *AI 2022: advances in artificial intelligence*. Editors H. Aziz, D. Corrêa, and T. French (Cham: Springer International Publishing), 733–746.

Tedrake, R. (2023) "Underactuated robotics: algorithms for walking," in *Running, swimming, flying, and manipulation (course notes for MIT 6.832)*.

Waharte, S., and Trigoni, N. (2010). "Supporting search and rescue operations with UAVs," in 2010 International Conference on Emerging Security Technologies (University of Oxford), 142–147. doi:10.1109/EST.2010.31

Wang, P., Zhu, M., and Shen, S. (2023). Environment transformer and policy optimization for model-based offline reinforcement learning *arXiv*. doi:10.48550/arXiv.2303.03811

Xu, J., Du, T., Foshey, M., Li, B., Zhu, B., Schulz, A., et al. (2019). Learning to fly: computational controller design for hybrid UAVs with reinforcement learning. *ACM Trans. Graph.* 38, 1–12. doi:10.1145/3306346.3322940

Yao, P., Xie, Z., and Ren, P. (2019). Optimal UAV route planning for coverage search of stationary target in river. *IEEE Trans. Control Syst. Technol.* 27, 822–829. doi:10.1109/TCST.2017.2781655

Yu, H., Wang, P., Wang, J., Ji, J., Zheng, Z., Tu, J., et al. (2023). Catch planner: catching high-speed targets in the flight. *IEEE/ASME Trans. Mechatronics* 28, 2387–2398. doi:10.1109/TMECH.2023.3286102

Yuksek, B., Umut Demirezen, M., Inalhan, G., and Tsourdos, A. (2021). Cooperative planning for an unmanned combat aerial vehicle fleet using reinforcement learning. *J. Aerosp. Inf. Syst.* 18, 739–750. doi:10.2514/1.I010961

Zhang, Y., Luo, B., Mukhopadhyay, A., Stojcsics, D., Elenius, D., Roy, A., et al. (2024). Shrinking pomcp: a framework for real-time UAV search and rescue. *2024 Int. Conf. Assur. Aut. (ICAA)*, 48–57. doi:10.1109/ICAA64256.2024.00016