Deep Reinforcement Learning with Dynamic Graphs for Adaptive Informative Path Planning

Apoorva Vashisth[1], Julius Rückin[2], Federico Magistri[2], Cyrill Stachniss[2], and Marija Popović[2]

***Abstract*—Autonomous robots are often employed for data collection due to their efficiency and low labour costs. A key task in robotic data acquisition is planning paths through an initially unknown environment to collect observations given platform-specific resource constraints, such as limited battery life. Adaptive online path planning in 3D environments is challenging due to the large set of valid actions and the presence of unknown occlusions. To address these issues, we propose a novel deep reinforcement learning approach for adaptively replanning robot paths to map targets of interest in unknown 3D environments. A key aspect of our approach is a dynamically constructed graph that restricts planning actions local to the robot, allowing us to react to newly discovered static obstacles and targets of interest. For replanning, we propose a new reward function that balances between exploring the unknown environment and exploiting online-discovered targets of interest. Our experiments show that our method enables more efficient target discovery compared to state-of-the-art learning and non-learning baselines. We also showcase our approach for orchard monitoring using an unmanned aerial vehicle in a photorealistic simulator. We open-source our code and model at: https://github.com/dmar-bonn/ipp-rl-3d.**

***Index Terms*—Motion and Path Planning, Reinforcement Learning, Robotics and Automation in Agriculture and Forestry**

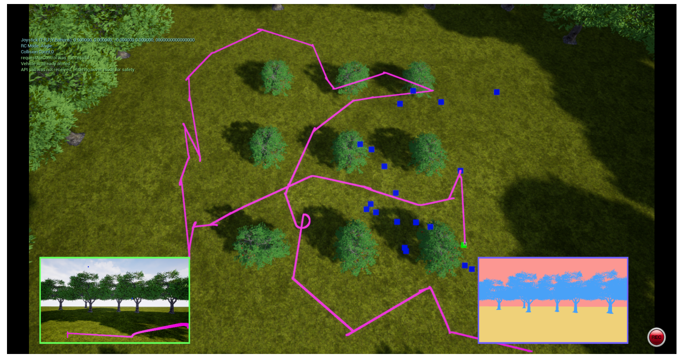


Fig 1: Our reinforcement learning approach for adaptive informative path planning applied in an orchard monitoring scenario using an unmanned aerial vehicle (UAV). Blue squares are candidate waypoints output by our planner, while the green square is the chosen next waypoint to visit. The inset windows show the onboard camera view and semantic segmentation for discovering apples. By planning collision-free paths for the UAV online, we maximise the number of apple fruits discovered under flight-length constraints.

I. Introduction

EFFICIENT data collection is a key requirement in many monitoring tasks, such as environmental mapping [1–5], precision agriculture [6–8], and exploration [9–11]. Autonomous robots are becoming popular tools for mobile sensing applications since they offer labour- and cost-effective alternatives to using conventional platforms, manual approaches [12], or static sensing methods [13]. A key challenge in this context is planning paths that maximise the information value of collected data in large environments with limited onboard resources, e.g. mission time or battery capacity.

In this work, our goal is to map a set of targets of interest in an initially unknown 3D environment using an unmanned aerial vehicle (UAV) with a unidirectional sensor as efficiently as possible. Possible applications for such a system are finding apples in an orchard, victims in a search and rescue scenario, or components in a warehouse. We cast this problem as the *informative path planning* problem, which aims to maximise the information value of obtained sensor observations subject to resource constraints, e.g. maximum path length or battery capacity. Our problem considers *adaptively* replanning robot paths to account for observations collected online. In our setting, adaptive replanning is challenging due to unknown view-dependent occlusions in the 4D action space, i.e. the UAV 3D position and yaw angle.

Classical approaches for this problem precompute a path before the mission starts [9, 14–16] without considering online-collected observations for replanning, likely leading to sub-optimal performance. In contrast, adaptive informative path planning approaches allow robots to replan their paths online based on newly collected data [4, 10, 17–28]. However, these methods are typically computationally inefficient in complex environments involving high-dimensional action spaces, such as UAV-based applications. Recently, approaches using deep reinforcement learning have been proposed for adaptive informative planning that outperform non-learning methods in various scenarios [29–32]. These methods are not directly applicable to our problem setting as they do not adjust the sensor orientation [29–31] or assume obstacle-free workspaces [30, 32]. Extending these approaches for obstacle avoidance is non-trivial, requiring updating the action space as the obstacles are discovered.

Manuscript received: February, 7, 2024; Revised May, 18, 2024; Accepted June, 14, 2024. This paper was recommended for publication by Editor A. Bera upon evaluation of the Associate Editor and Reviewers' comments. This work has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070 – 390732324 (PhenoRob).

[1]A.V. is with the Department of Mechanical Engineering, Indian Institute of Technology, Kharagpur, India. [2]J.R., F.M., C.S., and M.P. are with the Institute of Geodesy and Geoinformation, Cluster of Excellence PhenoRob, University of Bonn. M.P. is also with MAVLab, Faculty of Aerospace Engineering, TU Delft. C.S. is also with the University of Oxford and Lamarr Institute for Machine Learning and Artificial Intelligence, Germany. Corresponding: m.popovic@tudelft.nl.

Digital Object Identifier (DOI): see top of this page.

The main contribution of this paper is a deep reinforcement learning-based approach for adaptive replanning in unknown 3D environments to maximise the discovered targets of interest. A key aspect of our approach is a dynamically constructed graph representing the action space, which supports sequentially selecting the next best waypoint for the robot to visit. Our dynamic graph restricts planning to actions in the robot's local region at each timestep of the mission. In contrast to previous works [10, 17, 30] reasoning about static predefined action spaces, this enables us to plan informative collision-free paths, even without prior knowledge about the environment. Combining our dynamic graph with sequential decision-making through reinforcement learning allows us to plan long-horizon paths. To capture the adaptive informative planning objective, we propose a new reward function that encourages the robot to both explore unknown regions and exploit discovered targets. Fig. 1 exemplifies our approach applied on a UAV to discover fruits in an orchard.

In sum, we make the following three claims. First, our approach enables more efficiently discovering targets of interest compared to state-of-the-art non-learning-based and learning-based planning methods, including in previously unseen environments. Second, by adapting the action space online, our dynamic graph ensures collision-free navigation in initially unknown environments while performing on par with or outperforming static global representations. Third, our proposed reward function outperforms using purely exploratory rewards. We validate the performance of our approach in a realistic orchard monitoring UAV application.

II. Related Work

Informative path planning approaches are extensively applied in autonomous exploration and monitoring tasks. Classical methods [9, 14–16] either plan a path offline or optimise paths to cover the complete robot workspace [33]. These combinatorial methods do not allow for online replanning due to the large computational burden incurred when exhaustively evaluating all possible paths through the environment.

Generally, computational complexity is reduced by discretising the continuous action space by sampling and connecting candidate waypoints through platform-dependent paths. The robot visits only the sampled waypoints, traversing predefined paths formed by static global graphs representing the entire environment. Recent approaches [34–36] incrementally build the graph based on the current robot pose, reducing the replanning time. However, like classical approaches, these methods are non-adaptive to already collected observations for online replanning during a mission.

Adaptive informative path planning approaches [4, 10, 17–28] replan robot paths online and consider gathered observations to inform subsequent decision steps. Several studies apply evolutionary algorithms to optimise paths for UAVs [18, 23, 27] , autonomous surface vehicles [10], or autonomous ground vehicles [26] for adaptive replanning in a receding-horizon manner. Meera et al. [23] utilise covariance matrix adaptation evolution strategy (CMA-ES) to optimise the global path to map 2D target distributions using downward-facing cameras. Bouman et al. [26] maximise the environment coverage, planning in the robot's local region over 2D action spaces using omnidirectional sensors. Ercolani et al. [17] map gas distributions using nano aerial vehicles, separating path planning into global and local stages reasoning over clusters of waypoints. Similarly, Lim et al. [19] cluster waypoints by solving a group Steiner problem and frame UAV path planning as a travelling salesman problem over clusters. Oßwald et al. [28] combine globally optimal travelling salesman problem solutions on a coarse scale with effective local exploration adapting to the environment. Rückin et al. [4] and Mascarich et al. [21] derive information-theoretic planning objective to guide a UAV to cater for sensing uncertainty assuming obstacle-free workspaces. Schmid et al. [20] propose new techniques for node rewiring in sampling-based path planning strategies utilising a point sensor. A major limitation of adaptive replanning methods is the computationally expensive online evaluation of information values of many candidate paths in complex environments.

Recent studies combine neural networks and reinforcement learning to solve the informative path planning problem [29–32, 37]. Reinforcement learning-based solutions offer the benefits of computational efficiency at deployment and the ability to generalise to similar environments not seen during training. While Rückin et al. [32] combine Monte Carlo tree search with a convolutional neural network, Choi and Cielniak [31] consider advantages of paths planned by multiple low-level controllers. A further work by Cao et al. [30] propose an attention-based neural network to achieve context-aware path planning in 2D workspaces, whereas Wei and Zheng [29] use recurrent neural networks trained with Q-learning to plan the path. These methods are not directly applicable to our problem setup as they assume obstacle-free environments [29–32], map 2D information distributions [30, 32], or do not account for UAV yaw [29–31]. Our approach is most similar to Cao et al. [30], which plans over a probabilistic roadmap-based environment representation in obstacle-free 2D workspaces. A key difference with respect to their work is our reinforcement learning-based framework restricting robot's actions to local area, enabling planning in the presence of unknown obstacles. Moreover, our proposed reward function not only encourages exploratory behaviour, as done in previous studies [29, 30], but also incentivises exploiting newly collected information. We show that policies learned on our reward function outperform the ones learned on purely exploratory rewards.

III. Our approach

We propose a novel deep reinforcement learning-based informative path planning approach for maximising the number of discovered targets of interest in unknown 3D environments. Fig. 2 overviews our method. A key aspect of our approach is a graph restricting planning to actions in the robot's known local workspace, ensuring collision-free action transitions and allowing generation of collision-free paths. We call this environment representation a *dynamic graph* as it evolves to account for newly gathered observations. We estimate each action's utility value in the current dynamic

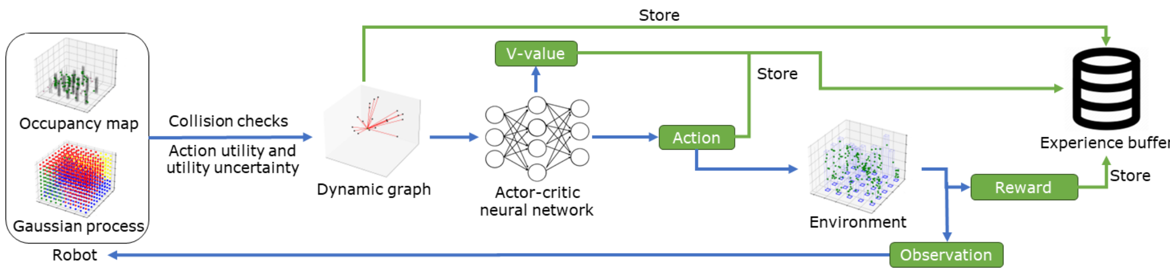


Fig 2: At each mission timestep t, our approach samples collision-free waypoints in the robot's local environment. These waypoints, with considered yaw directions, generate action nodes. Each action node is associated with utility value and uncertainty of the utility value, regressed from the Gaussian process, to generate the dynamic graph. Our actor-critic network uses the dynamic graph to output the robot's state value and predicts the next action to execute, which generates a reward and observations from the environment. Blue arrows indicate the robot control loop and green indicate variables stored in the experience buffer to train the actor-critic network via on-policy learning.

graph as the number of targets observed upon executing it and its corresponding uncertainty using a Gaussian process. At each timestep, our policy network outputs a probability distribution over the actions in the dynamic graph. We use the obtained observations to train the Gaussian process, update the occupancy map, and generate rewards reflecting the informative planning objective. We develop a new reward function that considers both the reduction in utility uncertainty and the number of observed targets. An experience buffer collects the robot's dynamic graph, sampled action, predicted state value, and reward over several training episodes to train the actor-critic network using on-policy reinforcement learning.

A. *Environment Modelling*

Our aim is to map the distribution of targets of interest in a 3D environment with unknown obstacles assumed to be static, non-moving. We use Gaussian processes to model the view-dependent number of observed targets. We also maintain an occupancy map to plan collision-free paths. Our mission budget $B \in \mathbb{R}^+$ is defined as the maximum cost of the traversed path. We define the robot workspace $\mathcal{A}$ as a set of actions $\mathbf{a}_i = [x_i, y_i, z_i, d_i]^\top$, where $x_i, y_i, z_i \in \mathbb{R}$ are the robot 3D position coordinates within environment bounds and $d_t \in \mathcal{D}$ is the yaw of the unidirectional onboard sensor, e.g., an RGB-D camera. The robot takes observations at each distance interval h as it travels. The set $\mathcal{D}$ denotes a user-defined set of possible robot yaw directions. During planning, at each mission timestep t, we plan over a set of L candidate actions in the set $\mathcal{A}_t \subseteq \mathcal{A}$, $|\mathcal{A}_t| = L$. The candidate actions are sampled uniformly at random from the C-neighbourhood around $\mathbf{a}_{t-1}$ within known free space and are checked for collision-free reachability along straight lines, where C is a constant specifying the extent of the robot's local region. In this work, we assume quasi-holonomic motion constraints for evaluating reachability of candidate actions. The reachability check can be adapted for non-holonomic motion constraints.

After executing an action $\mathbf{a}_t$, the robot observes a certain number of targets. Each action $\mathbf{a}_t \in \mathcal{A}$ in the robot workspace is associated with its utility value $u(\mathbf{a}_t) \in \mathbb{R}^+$ reflecting the number of targets observed. The number of observed targets is normalised by the total number of targets in the environment for stable policy training.

As the utility function $u : \mathcal{A} \rightarrow \mathbb{R}^+$ is (partially) unknown for actions $\mathbf{a}_t$, we need to estimate it. To this end, we utilise Gaussian processes widely used to estimate spatially correlated phenomena [10, 29, 30, 38]. We train a Gaussian process on the utility values of the actions executed in the past and exploit its utility estimates to inform the planning policy. The estimated variance allows our policy to consider the uncertainty over estimated utilities during planning.

A Gaussian process is characterised by a mean function $m(\mathbf{a}_i) \triangleq \mathbb{E}[u(\mathbf{a}_i)]$ and a covariance function $k(\mathbf{a}_i, \mathbf{a}_i') \triangleq \mathbb{E}[(u(\mathbf{a}_i) - m(\mathbf{a}_i))(u(\mathbf{a}_i') - m(\mathbf{a}_i'))]$ as $u(\mathbf{a}_i) \sim GP(m(\mathbf{a}_i), k(\mathbf{a}_i, \mathbf{a}_i'))$, where $\mathbb{E}[\cdot]$ is the expectation operator and $\mathbf{a}_i, \mathbf{a}_i' \in \mathcal{A}$. Hence, considering a set of candidate actions $\mathcal{A}_t$ at timestep t for which we wish to infer the utility, let the actions in set $\mathcal{A}_t$ correspond to a feature matrix $\mathbf{A}_t$, where each i^{th} row corresponds to the action vector $\mathbf{a}_t^i \in \mathcal{A}_t$. The set of all previously executed actions $\{\mathbf{a}_0, \mathbf{a}_2, \ldots, \mathbf{a}_{t-1}\}$ is represented by feature matrix $\mathbf{A}_{t-1}'$. Utility values corresponding to previously executed actions are represented by the vector $\mathbf{u}_{t-1}' = [u(\mathbf{a}_1), \ldots, u(\mathbf{a}_{t-1})]^\top$. We predict the utility values of candidate action set $\mathcal{A}_t$ by conditioning the Gaussian process on the observed utility values $\mathbf{u}_t \mid \mathbf{A}_{t-1}', \mathbf{u}_{t-1}', \mathbf{A}_t \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{P})$:

$$\begin{aligned}\boldsymbol{\mu}(\mathbf{A}_t, \mathbf{A}_{t-1}') = \mathbf{m}(\mathbf{A}_t) + \mathbf{K}(\mathbf{A}_t, \mathbf{A}_{t-1}')\,[\mathbf{K}(\mathbf{A}_{t-1}', \mathbf{A}_{t-1}') \\ + \sigma_n^2 \mathbf{I}]^{-1} \left(\mathbf{u}_{t-1}' - \mathbf{m}(\mathbf{A}_{t-1}')\right), \qquad (1)\end{aligned}$$

$$\begin{aligned}\mathbf{P}(\mathbf{A}_t, \mathbf{A}_{t-1}') = \mathbf{K}(\mathbf{A}_t, \mathbf{A}_t) - \mathbf{K}(\mathbf{A}_t, \mathbf{A}_{t-1}')[\mathbf{K}(\mathbf{A}_{t-1}', \\ \mathbf{A}_{t-1}') + \sigma_n^2 \mathbf{I}]^{-1} \mathbf{K}(\mathbf{A}_t, \mathbf{A}_{t-1}')^\top, \qquad (2)\end{aligned}$$

where $\sigma_n^2 \in \mathbb{R}^+$ is a hyperparameter describing the measurement noise variance, $\mathbf{I}$ is an $n \times n$ identity matrix where $n = t - 1$, and $\mathbf{K}(\cdot, \cdot)$ corresponds to the covariance matrix.

B. *Adaptive Informative Path Planning*

We model the path followed by the robot as a sequence of consecutively executed actions $\psi_0^T = (\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_T)$ where $\mathbf{a}_0$ denotes the start action, i.e. the initial robot pose, and $\mathbf{a}_T$ is the final action upon depletion of the budget B. The general informative path planning problem aims to find an optimal path ${\psi^*}_0^T \in \Psi$ in the space of all possible paths Ψ to optimise an information-theoretic objective function:

$$ {\psi^*}_0^T = \underset{\psi \in \Psi}{\operatorname{argmax}} \; I(\psi), \text{ s.t. } C(\psi) \leq B \,, \tag{3} $$

where $I : \psi_0^T \to \mathbb{R}^+$ is the information gained from observations obtained along path ψ_0^T, the cost function $C : \psi_0^T \to \mathbb{R}^+$ maps path ψ_0^T to its execution cost.

Our robot traverses a straight line between two consecutive actions. Observations are equidistantly collected along the path ψ_0^T at a frequency h and are used to update the Gaussian process and generate a reward. Hence, we model the informative path planning problem as a sequential decision-making process. As we aim to maximise the number of observed targets, we define a function $\nu : \mathcal{A} \times \Psi \to \mathbb{R}^+$ as the number of new targets observed upon executing an action $\mathbf{a}_t \in \mathcal{A}$ after following the path ψ_0^{t-1}. Note that information $\nu(\mathbf{a}_t, \psi_0^{t-1})$ and utility $u(\mathbf{a}_t)$ differ as the utility measures the number of all targets observed upon executing an action, whereas information considers only targets that were newly observed after executing the action. Hence, modelling $\nu(\mathbf{a}_t, \psi_0^{t-1})$ with a Gaussian process would require including the temporal variations of an action's utility value, increasing the Gaussian process and policy training complexity. We therefore choose to model utility with a Gaussian process, as it only depends on a single action $\mathbf{a}_t$.

We define the information obtained along a path as:

$$ I(\psi_0^T) = \sum_{t=1}^{T} \nu(\mathbf{a}_t, \psi_0^{t-1}) \,, \tag{4} $$

where we aim to plan a path ψ_0^T to maximise information I.

For informative planning, we leverage our Gaussian process defined in Sec. III-A to regress the utility and uncertainty associated with actions. We apply an upper confidence bound to the set of candidate actions $\mathcal{A}_t$ to obtain a subset of high-interest actions $\hat{\mathcal{A}}_t$ used in our reward function:

$$ \hat{\mathcal{A}}_t = \{\mathbf{a}_{i,t} \in \mathcal{A}_t \,|\, m(\mathbf{a}_{i,t}) + \beta k(\mathbf{a}_{i,t}, \mathbf{a}_{i,t}) \geq \mu_{th}\} \,, \tag{5} $$

where $m(\mathbf{a}_{i,t})$ and $k(\mathbf{a}_{i,t}, \mathbf{a}_{i,t})$ are the mean utility of action $\mathbf{a}_{i,t}$ and corresponding variance inferred from the Gaussian process. The parameter $\beta \in \mathbb{R}$ controls the confidence interval width and μ_{th} is a user-defined threshold.

We introduce a new reward function that balances exploring the environment and exploiting collected information. The information criteria in previous works [29, 30] consider environment exploration only. However, our problem considers discovering targets, therefore requiring a measure of information value in the reward. At each timestep t, the robot executes action $\mathbf{a}_t$, collects observations and receives a reward $r_t \in \mathbb{R}^+$. Our reward function consists of an exploration term $r_{e,t}$ and an information term $r_{u,t}$:

$$ \begin{aligned} r_t(\hat{\mathbf{A}}_t, \mathbf{A}'_{t-1}, \mathbf{A}'_t, \mathbf{a}_t, \psi_0^{t-1}) = \\ \alpha r_{e,t}(\hat{\mathbf{A}}_t, \mathbf{A}'_{t-1}, \mathbf{A}'_t) + \beta r_{u,t}(\mathbf{a}_t, \psi_0^{t-1}) \,, \end{aligned} \tag{6} $$

with:

$$ \begin{aligned} r_{e,t}(\hat{\mathbf{A}}_t, \mathbf{A}'_{t-1}, \mathbf{A}'_t) &= \frac{\mathrm{Tr}(\mathbf{P}(\hat{\mathbf{A}}_t, \mathbf{A}'_{t-1})) - \mathrm{Tr}(\mathbf{P}(\hat{\mathbf{A}}_t, \mathbf{A}'_t))}{\mathrm{Tr}(\mathbf{P}(\hat{\mathbf{A}}_t, \mathbf{A}'_{t-1}))} \,, \\ r_{u,t}(\mathbf{a}_t, \psi_0^{t-1}) &= \nu(\mathbf{a}_t, \psi_0^{t-1}) \,, \end{aligned} \tag{7} $$

where $\mathrm{Tr}(\cdot)$ is the trace operator of a matrix and α and β are constants used to trade off exploration and exploitation.

The variance reduction of the Gaussian process measures exploration $r_{e,t}$. To this end, we maximise the decrease in the covariance matrix trace following the A-optimal design criterion [39]. Scaling the reward by $\mathrm{Tr}(P(\hat{\mathbf{A}}_t, \mathbf{A}'_{t-1}))$ stabilises the actor-critic network training [30]. The term $r_{u,t}$ measures the new information gained after executing $\mathbf{a}_t$.

C. Dynamic Graph Representation

Adaptive informative path planning requires reasoning about the information distribution in the environment. We propose a dynamic graph that models the collision-free reachable action space and information distribution in the robot's neighbourhood by sampling actions as defined in Sec. III-A, as opposed to a static global non-obstacle-aware representation [10, 17, 30]. Our robot's policy relies on the representation of the current knowledge about the environment in the dynamic graph to predict next action Sec. III-D.

At each timestep t, we (re-)build a fully connected graph $\mathcal{G}_t = (\mathcal{N}_t, \mathcal{E}_t)$, where the node set $\mathcal{N}_t$ is the set of candidate actions $\mathcal{A}_t$ defined in Sec. III-A, to account for newly gathered observations. We randomly sample K candidate positions $[x_i, y_i, z_i]^\top$ within a C-neighbourhood of the current robot position and create $L = K \times |\mathcal{D}|$ nodes by associating each position with possible yaws $\mathcal{D}$. The edge set $\mathcal{E}_t$ connects each pair of actions such that $e_{i,j,t} = (i, j, c_{i,j}) \in \mathcal{E}_t$, where the cost $c_{i,j}$ is the edge weight, $i, j \leq L$, and $i \neq j$. The edge set $\mathcal{E}_t$ and the cost $c_{i,j}$ are based on the robot's motion constraints, defining the path traversed from action $\mathbf{a}_i$ to $\mathbf{a}_j$. In this work, we consider a holonomic UAV travelling between two actions on a straight line, hence the cost $c_{i,j}$ is defined as the sum of the actions' Euclidean distance and a small constant cost C_s if the robot yaw changes.

To better inform the planning policy, we leverage our Gaussian process to create node features utilised in our actor-critic network. At each timestep t, the node feature matrix $\mathbf{M}_t$ of graph $\mathcal{G}_t$ consists of the robot's candidate actions and the mean and variance values queried from the Gaussian process. The i^{th} row of $\mathbf{M}_t$ relates to the i^{th} action:

$$ \mathbf{M}_t(i) = [\mathbf{a}_{i,t},\, m(\mathbf{a}_{i,t}),\, k(\mathbf{a}_{i,t}, \mathbf{a}_{i,t})] \,, \tag{8} $$

where $\mathbf{a}_{i,t} = [x_{i,t},\, y_{i,t},\, z_{i,t},\, d_{i,t}]^\top$, and $m(\mathbf{a}_{i,t})$ and $k(\mathbf{a}_{i,t}, \mathbf{a}_{i,t})$ are the regressed action's utility and variance.

D. Actor-Critic Neural Network for Reinforcement Learning

We exploit the dynamic graph to model collision-free actions for the planning policy to reason about and represent the current knowledge of the information distribution in the environment. As the Gaussian process only predicts the utility of greedily executing a single next action, we use reinforcement learning to train our policy for informative path planning over long-horizon paths.

We use an attention-based neural network to parameterise our stochastic planning policy $\pi(\mathcal{G}_t, \psi_0^{t-1}, B_r, \mu_{th}) \in [0,1]^L$ that predicts a probability distribution over all L actions $\mathcal{A}_t$ based on the current dynamic graph $\mathcal{G}_t$, previously executed path ψ_0^{t-1}, remaining budget B_r, and the mean threshold μ_{th} defining actions of interest in Eq. (5). We follow the network structure proposed by Cao et al. [30] consisting of an encoder and a decoder module. The attention-based encoder learns the dependencies between actions in $\mathcal{G}_t$. We condition the learned actions' latent dependencies on a planning state consisting of previously executed actions ψ_0^{t-1}, the remaining budget B_r, and the threshold μ_{th}. A budget mask filters out actions not reachable within the remaining budget. Based on the conditioned latent action dependencies, a decoder outputs a probabilistic policy reasoning over all actions in the dynamic graph. During training, the decoder also estimates the value function $V(\mathcal{G}_t, \psi_0^{t-1}, B_r, \mu_{th}) \in \mathbb{R}$ following the current policy $a_t \sim \pi(\mathcal{G}_t, \psi_0^{t-1}, B_r, \mu_{th})$. The estimated values, sampled actions, dynamic graphs, planning states, and rewards are stored in the experience buffer utilised to train the policy with an on-policy actor-critic reinforcement algorithm. In this work, we use proximal policy optimisation [40]. During deployment, at each time step t, we choose the most informative action from $\pi(\mathcal{G}_t, \psi_0^{t-1}, B_r, \mu_{th})$.

IV. Experimental Results

We experimentally validate our three claims on the task of UAV-based fruit monitoring in apple orchards. First, our approach enables more efficiently discovering targets of interest compared to non-learning baselines and learning-based methods. Second, our dynamic graph action space enables collision-free path planning in unknown environments while performing on par with state-of-the-art static global graph representations. Third, our new reward function more effectively manages the exploration-exploitation trade-off compared to using purely exploratory rewards, leading to more efficient targeted mapping. We demonstrate the performance of our approach in a realistic orchard simulation, showcasing its applicability for a practical monitoring task in a previously unseen environment.

A. Experimental Setup

Environment. Our environment consists of trees and fruits bounded in a scale-agnostic unit cube. We maintain an occupancy map with $50 \times 50 \times 50$ voxels. During test phase, trees are generated at random positions in the environment but are arranged in a regularly spaced 5×5 array during training. Fig. 3 shows examples of a training and a testing environment. In both cases, fruits are attached to generated trees at random positions. The occupancy grid map of the environment is initialised as unknown space and is updated via sensor observations with free space, observed fruits, and trees. For each observation, the utility value is used to train the Gaussian process detailed in Sec. III-A. We tune the hyperparameters in a small representative environment using the Matérn $1/2$ kernel with $\mu_{th} = 0.4$ and $\beta = 1$ in Eq. (5). For our reward function defined in Eq. (3), we set $\alpha = 1.0$ and $\delta = 0.01$ to keep values of both terms numerically similar to balance between exploration and exploitation.

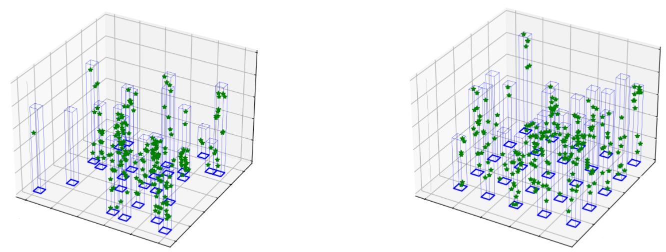


Fig 3: Examples of testing and training orchard environments used in our experiments. *Left*: Testing environment with trees placed at random locations. *Right*: Training environment with trees placed in a regular square array. Thin and dark blue lines represent tree outlines and tree bases, respectively. Green stars indicate fruits.

We consider a UAV platform with an onboard RGB-D camera of $90°$ field of view (FoV). Since the confidence in discovered targets decreases with distance, we constrain the maximum detection range to 24% of the environment size. The UAV can choose between yaw angles of $[0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}]$ rad at the current altitude. Note that our method can be easily extended to finer discretisations.

Training An episode consists of a UAV mission with a budget B. We train in a grid-based environment as a randomly initialised policy learns efficiently from this structure and transfers to randomised test environments. The total number of fruits varies between 200 and 250. We fix the number of positions K in the dynamic graph and set the initial UAV pose to $[0, 0, 0, \frac{\pi}{2}]^\top$. To keep our policy scale-agnostic, we normalise the robot's internal environment representation and action coordinates. Hence, the budget B is unitless and randomly generated for each episode within the range $[7, 9]$. We fuse an observation into the occupancy map and Gaussian process each time the UAV travels 0.2 units from the position of the previous observation.

We terminate an episode if the maximum number of executed actions exceeds 256. To speed up training, we run 12 environments in parallel. The policy is trained for 8 epochs using a batch size of 64 and the Adam optimiser with a learning rate of 10^{-4}, which decays by a factor of 0.96 each 32 optimisation steps. The policy gradient epsilon-clip parameter is set to 0.2. Our model is trained on a workstation equipped with an Intel(R) Xeon(R) W-2133 CPU @ 3.60GHz and one NVIDIA Quadro RTX 5000 GPU. Our policy is trained for $\sim 440,000$ environment interactions.

B. Comparison Against Baselines

The first set of experiments shows that our dynamic graph-based reinforcement learning approach outperforms state-of-the-art baselines. We generate 25 test environments corresponding to different random seeds and run 20 trials on each environment instance with a budget of 10 units.

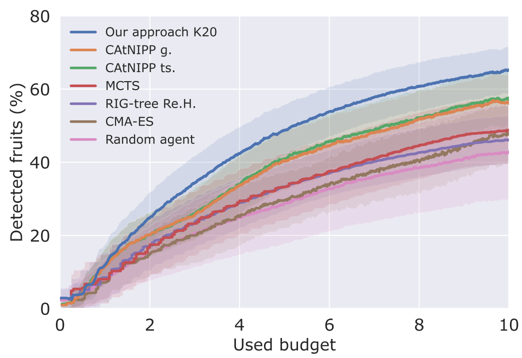


Fig 4: Comparison of our approach against baselines in a UAV-based fruit monitoring scenario. Solid lines indicate means over 500 trials and shaded regions show standard deviations. In our approach, using our exploration-exploitation reward function with a dynamic graph action space for reinforcement learning enables more efficiently discovering targets of interest (fruit) during a mission.

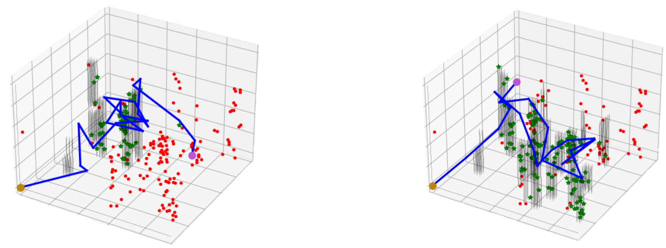
TABLE I: Comparison of our approach against baselines in a UAV-based fruit monitoring scenario. Our dynamic graph-based reinforcement learning approach outperforms learning and non-learning baselines while maintaining low replanning time.

Baseline	% targets	Time (s)
Our approach $K = 20$	65.49 ± 6.23	1.58
CAtNIPP (ts.)	57.14 ± 7.60	44.51
CAtNIPP (g.)	56.49 ± 7.65	1.28
MCTS	50.15 ± 8.98	218.83
CMA-ES	48.72 ± 8.36	165.41
RIG-tree (Re.H.)	47.43 ± 6.51	80.27
Random agent	43.91 ± 12.64	0.28

For our approach, we consider $K = 20$ sampled waypoints, $L = 80$ nodes in the dynamic graph, and the reward function described in Eq. (3). The learning-based baseline for evaluation is CAtNIPP [30], the state-of-the-art approach closest to our work which uses global graph-based planning. Since CAtNIPP considers obstacle-free environments and modifying its global graph to account for unknown obstacles is a non-trivial task, we allow the UAV to pass through obstacles to ensure a fair comparison.

We compare against: (i) CAtNIPP with a zero-shot policy (CAtNIPP g.) where the highest probability action is executed; (ii) CAtNIPP with a trajectory sampling policy [30] (CAtNIPP ts.) where four 8-step paths are planned and the path with highest uncertainty reduction is executed for 3 steps; (iii) a random policy on $K = 20$ dynamic graph construction (random agent); (iv) non-learning rapidly exploring random information gathering trees [34] applied in a receding-horizon manner (RIG-tree Re.H.); (v) non-learning Monte Carlo Tree Search [41] (MCTS); and (vi) non-learning Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [42]. To evaluate planning performance, we measure the percentage of discovered targets of interest (apples) during the test. We also report average replanning time per step.

Fig. 4 and Table I illustrate our results. Our approach outperforms all baselines by a significant margin. This can be attributed to the new reward function that balances between exploring the environment and exploiting collected information.



(a) CAtNIPP (g.) (b) Our approach

Fig 5: Comparison of paths planned by (a) the global graph-based CAtNIPP [30] baseline and (b) our dynamic graph-based reinforcement learning approach with $K = 20$ in a fruit monitoring scenario. The blue line shows the executed UAV path, with the brown and pink circles indicating start and end positions. Red dots are targets not yet observed, while green stars are observed targets.

TABLE II: Graph structure ablation study. All methods use our proposed exploration-exploitation reward. The dynamic graph structure performs slightly better than the global representation of CAtNIPP.

Graph Structure	% targets	Time (s)
$K = 10$	60.85 ± 7.51	1.40
$K = 20$	65.49 ± 6.23	1.58
$K = 30$	64.20 ± 6.62	1.99
$K = 40$	66.53 ± 6.28	2.75
$K = 50$	64.20 ± 6.80	3.80
CAtNIPP [30]	63.27 ± 6.89	1.28

Both CAtNIPP variants perform worse than our method since they only focus on utility variance reduction. In Table I, the approaches using reinforcement learning require significantly less replanning time than non-learning methods, justifying the use of learning-based strategies. Our proposed approach is more compute-efficient than CAtNIPP (ts.), and almost as efficient as CAtNIPP (g.), which facilitates its deployment in real-world scenarios.

Fig. 5 qualitatively compares the paths planned by our approach and CAtNIPP g. for the ground truth environment illustrated in Sec. IV-A. The visualisations correspond to paths executed at 50% of the budget. Our approach favours actions that discover more targets. This is because CAtNIPP considers a purely exploratory objective, assuming a continuous distribution of utility, which leads to re-observing the high-interest regions in an oscillatory manner. Our approach does not encounter this issue. Since we reduce both uncertainty and maximise the number of discovered targets using our reward function, we obtain a more widespread distribution of observations in the environment.

C. Ablation Studies

Next, we systematically study the impact of our dynamic graph action space and proposed reward function on the informativeness of the planned paths to demonstrate their benefits. Our test environment is the same as in Sec. IV-B. We compare our approach against CAtNIPP [30], which uses a static graph action space based on probabilistic roadmaps and a purely exploratory reward function.

Graph Structure. To investigate the influence of the number of sampled waypoints K on planning performance,

TABLE III: Reward function ablation study. Both dynamic graph- and global graph-based policies trained on our reward function outperform those learned on a purely exploratory reward.

Model and reward function	% targets
$K = 20$, exploration	54.75 ± 7.66
$K = 20$, our reward	65.49 ± 6.23
CAtNIPP [30], our reward	63.27 ± 6.89
CAtNIPP [30], exploration	56.52 ± 8.29

we compare our approach using dynamic graphs with $K \in \{10, 20, 30, 40, 50\}$ against CAtNIPP. Here, we use our new reward function that combines both exploitation of obtained information and exploration of unknown regions, as described in Eq. (6), ensuring that the performance variations can be attributed to the graph structure alone. Table II summarises the results. We observe performance improvements from $K = 10$ to $K = 20$ and similar performance for $K \geq 20$ at the cost of increasing computation time. The performance of our dynamic graph structure with $K \geq 20$ is slightly better than the global graph of CAtNIPP. Hence, our dynamic graph can actively account for unknown obstacles while performing similarly to, or better than, the global graph structure.

Reward Function. Next, we investigate the effects of training a policy on our reward function against a purely exploratory reward function. We compare our dynamic graph-based approach trained on $K = 20$ waypoints and 80 action nodes against CAtNIPP trained on $K = 200$ waypoints and 800 action nodes. We tune the hyperparameters for best performance and consider two variants of each method with the different reward functions. For the purely exploratory reward, we set $\alpha = 1$ and $\delta = 0$ in Eq. (6). Table III summarises the results. Both our dynamic graph and the CAtNIPP global graph trained on our new reward function outperform the corresponding policies trained using purely exploratory rewards. This confirms that learning from exploration rewards alone cannot guide the robot to adaptively focus on targets of interest since it incentivises actions that reduce overall utility variance. In contrast, our reward function incorporating both uncertainty reduction and targeted information gathering yields better target discovery performance as it allows the policy to learn the trade-off between exploration and exploitation. The new reward function benefits both our dynamic graph and the global graph, showing its general applicability for different informative planning algorithms.

D. Realistic Simulation

We demonstrate the applicability of our dynamic graph-based reinforcement learning approach with $K = 20$ sampled waypoints in an orchard environment simulator created with Unreal Engine and AirSim. The Airsim simulator resembles real-world UAV dynamics, while Unreal Engine provides photorealistic imagery. Our apple orchard environment is bounded by a 95 m ×95 m ×18 m cuboid with 9 trees arranged in a 3×3 array and a total of 225 red apples at random locations on the trees as illustrated in Fig. 1. We assume perfect localisation and use ground truth apple object discovery. The UAV moves at a maximum speed of 2 m/s.

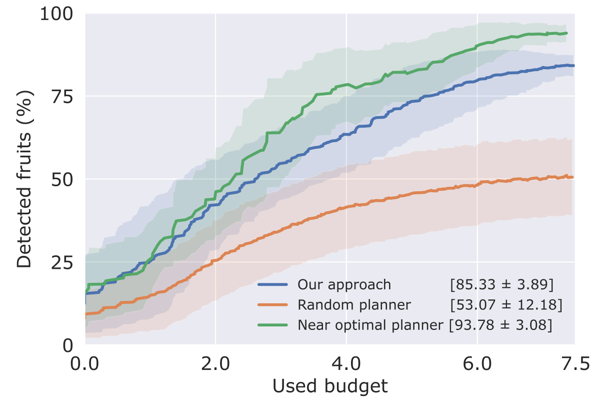


Fig 6: Comparison of our reinforcement learning-based approach against baselines in a photorealistic fruit monitoring simulator. Solid lines indicate means over 10 trials and shaded regions show standard deviations. Our approach performs almost as well as the near-optimal baseline, despite being trained in different environments and not relying on ground truth knowledge.

We compare our approach trained in the synthetic simulation described in Sec. IV-A against (i) a random planner over our $K = 20$ dynamic graph and (ii) a near-optimal planner to reflect performance upper bound using the metric of percentage discovered fruits. We record the coordinates of discovered fruits to ensure that the same fruit is not counted multiple times. We design the near-optimal planner to exploit ground truth information of the environment, such as tree coordinates, size, and best altitude, to generate a coverage-like path for observing maximum fruits. We run several instances of this planner and choose the three best-performing paths to compare against our approach. For our planner and the random planner, results are reported over 10 trials in the environment with a mission budget of 7.5 units.

Fig. 6 compares the three planners. Our approach outperforms non-informative random planning. The near-optimal planner performs best since it exploits ground truth knowledge to avoid viewpoint-dependent occlusions. However, our approach reaches similar performance without relying on any prior knowledge, making it suitable for unknown fruit distributions. Fig. 1 visualises the path executed by our planner. These findings support the applicability of our method on a UAV platform in a practical monitoring scenario.

V. Conclusion and Future Work

We present a deep reinforcement learning approach for adaptively discovering targets of interest in unknown 3D environments. A key aspect of our method is a dynamic graph constructing a detailed environment representation to constrain planning in the robot's local region. We also present a new reward function enabling our learned policy to balance exploring the environment and exploiting obtained information. Our experimental results support our three claims: (i) our approach outperforms the state-of-the-art learning-based approaches and non-learning baselines in environments unseen during training; (ii) our dynamic graph approach leads to performance on par, or better, than static global graph based state-of-the-art methods; (iii) our new reward function outperforms a purely exploratory reward function. We validate our approach in a UAV-based fruit monitoring scenario to demonstrate its practical applicability. Future work includes developing advanced sampling strategies, considering dynamic obstacles, and transferring our policy to a real robot under localisation and perception uncertainty.

References

[1] A. Dhariwal, G. S. Sukhatme, and A. A. Requicha, "Bacterium-inspired robots for environmental monitoring," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2004.

[2] M. Duarte, J. Gomes, V. Costa, T. Rodrigues, F. Silva, V. Lobo, M. M. Marques, S. M. Oliveira, and A. L. Christensen, "Application of swarm robotics systems to marine environmental monitoring," in *Proc. of OCEANS MTS/IEEE Conf. and Exhibition*, 2016.

[3] R. Marchant and F. Ramos, "Bayesian optimisation for intelligent environmental monitoring," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[4] J. Rückin, F. Magistri, C. Stachniss, and M. Popović, "An Informative Path Planning Framework for Active Learning in UAV-Based Semantic Mapping," *IEEE Trans. on Robotics (TRO)*, vol. 39, no. 6, pp. 4279–4296, 2023.

[5] Y. Wang, Y. Wang, Y. Cao, and G. Sartoretti, "Spatio-Temporal Attention Network for Persistent Monitoring of Multiple Mobile Targets," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2023.

[6] F. Magistri, N. Chebrolu, and C. Stachniss, "Segmentation-based 4D registration of plants point clouds for phenotyping," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[7] C. Carbone, D. Albani, F. Magistri, D. Ognibene, C. Stachniss, G. Kootstra, D. Nardi, and V. Trianni, "Monitoring and mapping of crop fields with uav swarms based on information gain," in *Distributed Autonomous Robotic Systems*, 2022.

[8] F. Magistri, D. Nardi, and V. Trianni, "Using prior information to improve crop/weed classification by MAV swarms," in *Proc. of the Int. Micro Air Vehicle Competition and Conf.*, 2019.

[9] J. Binney, A. Krause, and G. S. Sukhatme, "Informative path planning for an autonomous underwater vehicle," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2010.

[10] G. Hitz, E. Galceran, M.-È. Garneau, F. Pomerleau, and R. Siegwart, "Adaptive continuous-space informative path planning for online environmental monitoring," *Journal of Field Robotics (JFR)*, vol. 34, no. 8, pp. 1427–1449, 2017.

[11] Y. Cao, T. Hou, Y. Wang, X. Yi, and G. Sartoretti, "ARiADNE: A Reinforcement learning approach using Attention-based Deep Networks for Exploration," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2023.

[12] J. Su, X. Zhu, S. Li, and W.-H. Chen, "AI meets UAVs: A survey on AI empowered UAV perception systems for precision agriculture," *Neurocomputing*, vol. 518, pp. 242–270, 2022.

[13] F. M. Al-Turjman, H. S. Hassanein, and M. A. Ibnkahla, "Efficient deployment of wireless sensor networks targeting environment monitoring applications," *Computer Communications*, vol. 36, no. 2, pp. 135–148, 2013.

[14] J. Binney and G. S. Sukhatme, "Branch and bound for informative path planning," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2012.

[15] S. Arora and S. Scherer, "Randomized algorithm for informative path planning with budget constraints," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2017.

[16] R. Marchant and F. Ramos, "Bayesian optimisation for informative continuous path planning," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2014.

[17] C. Ercolani, L. Tang, A. A. Humne, and A. Martinoli, "Clustering and informative path planning for 3D gas distribution mapping: Algorithms and performance evaluation," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 5310–5317, 2022.

[18] M. Popović, T. Vidal-Calleja, G. Hitz, J. J. Chung, I. Sa, R. Siegwart, and J. Nieto, "An informative path planning framework for UAV-based terrain monitoring," *Autonomous Robots*, vol. 44, pp. 889–911, 2020.

[19] Z. W. Lim, D. Hsu, and W. S. Lee, "Adaptive informative path planning in metric spaces," *Int. Journal of Robotics Research (IJRR)*, vol. 35, no. 5, pp. 585–598, 2016.

[20] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An efficient sampling-based method for online informative path planning in unknown environments," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, pp. 1500–1507, 2020.

[21] F. Mascarich, C. Papachristos, T. Wilson, and K. Alexis, "Distributed radiation field estimation and informative path planning for nuclear environment characterization," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2019.

[22] H. Zhu, J. J. Chung, N. R. Lawrance, R. Siegwart, and J. Alonso-Mora, "Online Informative Path Planning for Active Information Gathering of a 3D Surface," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2021.

[23] A. A. Meera, M. Popović, A. Millane, and R. Siegwart, "Obstacle-aware Adaptive Informative Path Planning for UAV-based Target Search," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2019.

[24] C. Rhodes, C. Liu, and W.-H. Chen, "Informative path planning for gas distribution mapping in cluttered environments," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[25] F. Stache, J. Westheider, F. Magistri, C. Stachniss, and M. Popović, "Adaptive path planning for uavs for multi-resolution semantic segmentation," *Journal on Robotics and Autonomous Systems (RAS)*, vol. 159, p. 104288, 2023.

[26] A. Bouman, J. Ott, S.-K. Kim, K. Chen, M. J. Kochenderfer, B. Lopez, A.-a. Agha-mohammadi, and J. Burdick, "Adaptive coverage path planning for efficient exploration of unknown environments," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2022*.

[27] E. Palazzolo and C. Stachniss, "Effective exploration for mavs based on the expected information gain," *Drones*, vol. 2, no. 1, p. 9, 2018.

[28] S. Oßwald, M. Bennewitz, W. Burgard, and C. Stachniss, "Speeding-up robot exploration by exploiting background information," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 716–723, 2016.

[29] Y. Wei and R. Zheng, "Informative path planning for mobile sensing with reinforcement learning," in *Proc. of the IEEE Conf. on Computer Communications*, 2020.

[30] Y. Cao, Y. Wang, A. Vashisth, H. Fan, and G. A. Sartoretti, "CAtNIPP: Context-aware attention-based network for informative path planning," in *Proc. of the Conf. on Robot Learning (CoRL)*, 2023.

[31] T. Choi and G. Cielniak, "Adaptive selection of informative path planning strategies via reinforcement learning," in *Proc. of the Europ. Conf. on Mobile Robotics (ECMR)*, 2021.

[32] J. Rückin, L. Jin, and M. Popović, "Adaptive informative path planning using deep reinforcement learning for UAV-based active sensing," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2022.

[33] W. Jing, D. Deng, Z. Xiao, Y. Liu, and K. Shimada, "Coverage path planning using path primitive sampling and primitive coverage graph for visual inspection," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[34] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *Int. Journal of Robotics Research (IJRR)*, vol. 33, no. 9, pp. 1271–1287, 2014.

[35] C. Wang, J. Cheng, W. Chi, T. Yan, and M. Q.-H. Meng, "Semantic-aware informative path planning for efficient object search using mobile robot," in *Proc. of the IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, 2019.

[36] B. Moon, S. Chatterjee, and S. Scherer, "TIGRIS: An Informed Sampling-based Algorithm for Informative Path Planning," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

[37] X. Zeng, T. Zaenker, and M. Bennewitz, "Deep Reinforcement Learning for Next-Best-View Planning in Agricultural Applications," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2022.

[38] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[39] R. Sim and N. Roy, "Global A-Optimal Robot Exploration in SLAM," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[40] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[41] J. Ott, E. Balaban, and M. J. Kochenderfer, "Sequential bayesian optimization for adaptive informative path planning with multimodal sensing," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA), 2023*.

[42] M. Popović, T. Vidal-Calleja, G. Hitz, I. Sa, R. Siegwart, and J. Nieto, "Multiresolution mapping and informative path planning for uav-based terrain monitoring," in *Proc. of the IEEE Int. Conf. on Intelligent Robots and Systems (IROS), 2017*.