OffRIPP: Offline RL-based Informative Path Planning

Srikar Babu Gadipudi SRIKARBABU@BERKELEY.EDU
University of California, Berkeley, CA 94720
Srujan Deolasee SRUJAN.DEOLASEE@GENERALROBOTICS.COMPANY
General Robotics, Redmond, WA 98052, USA
Siva Kailas SKAILAS3@GATECH.EDU
Georgia Institute of Technology, Atlanta, GA 30332, USA
Wenhao Luo WENHAO@UIC.EDU
University of Illinois Chicago, Chicago, IL 60607, USA
Katia Sycara SYCARA@ANDREW.CMU.EDU
Woojun Kim WOOJUNK@ANDREW.CMU.EDU
Carnegie Mellon University, Pittsburgh, PA 15213, USA

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

Informative path planning (IPP) is a crucial task in robotics, where an agent designs paths to gather valuable information about a target environment while adhering to resource constraints. Reinforcement learning (RL) has been shown to be effective for IPP; however, it typically requires online interaction with the environment, which is risky and expensive in practice. To address this challenge, we propose an offline RL-based IPP framework that optimizes information gain without requiring real-time interaction during training, offering safety and cost-efficiency by avoiding additional interactions, while achieving superior performance and fast computation during execution. Our framework leverages batch-constrained RL to mitigate extrapolation errors, enabling the agent to learn from pre-collected datasets generated by arbitrary algorithms. We validate the framework through evaluations on diverse offline datasets and real-world experiments. The numerical results show that our framework outperforms baseline methods, demonstrating its effectiveness.

Keywords: Informative Path Planning; Offline Reinforcement Learning;

1. INTRODUCTION

Informative path planning (IPP) is a critical problem in robotics and autonomous systems, where the goal is to design a path that enables an agent to acquire valuable information about a quantity of interest within an environment while adhering to resource constraints, such as limited battery capacity. IPP has numerous applications, including environmental monitoring (Hitz et al., 2017), search and rescue (Meera et al., 2019), and precision agriculture (Popović et al., 2020). The central challenge is to plan trajectories that effectively approximate the latent spatial field or interest map.

Traditional IPP methods (Karaman and Frazzoli, 2011; Arora and Scherer, 2017) rely on sampling-based techniques that, while effective, can be computationally expensive and difficult to scale to large or complex environments. As a result, recent work has begun exploring reinforcement learning (RL) as a data-driven alternative for IPP (Cao et al., 2023; Vashisth et al., 2024), where an agent learns decision policies through interactions with the underlying dynamical environment. However, standard RL methods typically require extensive real-time interactions, making training costly and potentially unsafe for many physical systems. Moreover, in many settings, domain-relevant historical datasets already exist. This motivates offline RL, which learns policies directly from pre-collected trajectory data without additional online interactions (Kumar et al., 2020).

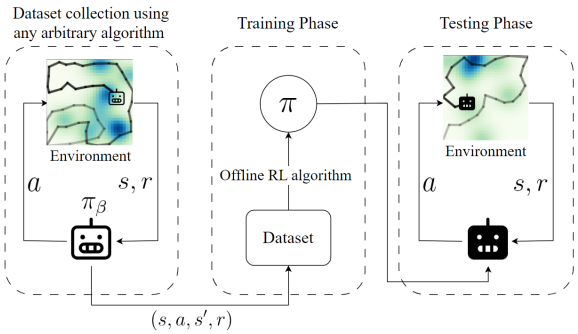


Figure 1: Overview of the flow for the proposed framework: training an RL policy using a dataset generated by arbitrary algorithms, followed by deployment in the test environment.

In this paper, we propose `OffRIPP`, an **Off**line **RL**-based **IPP** framework that plans paths to maximize information gain without requiring real-time interactions with the environment, relying solely on pre-collected datasets. These datasets may include optimal and sub-optimal trajectories, yet our goal is to learn a policy that maximizes information gain regardless of dataset quality. Specifically, we train an RL agent to optimize information acquisition under a constrained budget using only offline trajectory data. To achieve this, we adopt batch-constrained reinforcement learning (Fujimoto et al., 2019) to mitigate extrapolation errors caused by distribution mismatch between the dataset and the learned policy. That is, we approximate the dataset's behavior policy and use it to constrain the learned policy during training. Once training is complete, the learned policy can be directly deployed in the test environment. The overall pipeline is illustrated in Fig. 1. Our framework provides an efficient and safe solution for data-driven decision making over dynamical systems in the context of informative path planning, combining the advantages of RL—fast online planning and strong performance—with the benefits of traditional approaches that do not require additional environment interaction.

We validate our approach using extensive evaluations on diverse pre-collected datasets and real-world experiments, demonstrating that our offline RL framework improves IPP performance without requiring additional environment interactions. Specifically, we evaluate `OffRIPP` on two existing RL-based IPP algorithms in two environments: a 2D light-intensity task and a 3D fruit identification task. For each environment, we consider three types of datasets: expert trajectories, medium-quality trajectories collected by online RL, and a greedy planning dataset generated by a simple non-learning heuristic. The numerical results show that `OffRIPP` outperforms baseline methods including traditional approaches, online RL-based algorithms trained offline, and behavior cloning across all environments and dataset types, demonstrating its robustness and effectiveness. Moreover, `OffRIPP` requires only approximately 10% of the execution time of a traditional non-learning baseline, highlighting its computational practicality. Finally, we demonstrate that `OffRIPP`, trained solely in simulation, can be successfully deployed on a physical robot platform.

Our main contributions are summarized as follows: (i) To the best of our knowledge, this is the first work to leverage offline RL for informative path planning, training policies solely from pre-collected datasets without additional environment interaction. (ii) Our approach is modular and can be combined with any online RL-based IPP method; we demonstrate this using CAtNIPP (Cao et al., 2023) and another RL-based IPP method (Vashisth et al., 2024). (iii) We demonstrate improvements in solution quality and planning time, and additionally study the effect of dataset quality.

2. Background and Related works

2.1. Offline Reinforcement Learning

RL trains an agent through interactions with an environment. At each time step t, the agent's policy π selects an action a_t based on the current state s_t. The environment then provides a re-

ward r_t and transitions to the next state s_{t+1}. By repeating this process in an online learning fashion, the policy aims to maximize the expected return $\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]$. To evaluate and improve the policy, most RL algorithms estimate the value function under the current policy, such as $Q^\pi(s,a) = \mathbb{E}_\pi[\sum_{t=0}^{\infty}\gamma^t r_t \mid s,a]$, which denotes the expected return after taking action a in state s and then following π thereafter. A common approach for approximating the value function is temporal-difference (TD) learning (Tesauro et al., 1995), which can be written as

$$Q^\pi(s,a) \leftarrow \mathbb{E}_{s'}\left[r + \gamma Q^\pi(s', \pi(s'))\right], \tag{1}$$

where s' denotes the next state. Since the policy is improved to maximize the estimated value, accurate value estimation is critical for effective policy learning.

Offline RL trains an agent on a fixed dataset collected from arbitrary policies, without any interaction with the environment during training (Kim et al., 2024b; Kumar et al., 2020; Kim et al., 2024a). A major challenge in offline RL is the extrapolation error that arises from distribution mismatch between the dataset and the state-action distribution induced by the learned policy, leading to inaccurate value estimates (Fujimoto et al., 2019). For example, in TD learning, if a state-action pair $(s', \pi(s'))$ is not present in the dataset, the corresponding value estimate can be highly inaccurate. In other words, the expectation in Eq. 1 suffers from a large approximation error, which accumulates in the estimate of $Q^\pi(s,a)$. Thus, the larger the mismatch between the dataset and the learned policy's distribution, the greater the risk of compounding approximation error. To mitigate extrapolation error, offline RL algorithms encourage the learned policy to remain close to the behavior policy that generated the dataset (Fujimoto et al., 2019; Kostrikov et al., 2021; Kumar et al., 2020). For example, Conservative Q-learning (CQL) penalizes Q-values associated with actions not observed in the dataset (Kumar et al., 2020), discouraging the selection of out-of-distribution actions. Batch-Constrained Q-learning models the behavior policy directly and constrains the learned policy to select only actions observed in the dataset (Fujimoto et al., 2019).

2.2. Informative Path Planning

The goal of IPP is to find a trajectory ψ^* that maximizes the information gain within the given budget (Rückin et al., 2023; Kailas et al., 2023), and the objective function is written as

$$\psi^* = \arg\max_{\psi} I(\psi), \text{ s.t. } C(\psi) \leq B, \tag{2}$$

where $I : \psi \to \mathbb{R}^+$ and $C : \psi \to \mathbb{R}^+$ denote the information gain and the cost along ψ, and $B \in \mathbb{R}^+$ is the budget. The definition of information gain depends on the domain, e.g., fire intensity in wildfire monitoring. Traditional IPP algorithms utilize sampling-based methods (Karaman and Frazzoli, 2011; Jones et al., 2013; Yoo et al., 2016; Hitz et al., 2017), but they require significant computational resources, which limits their practicality in real-world settings.

RL-based IPP: RL has been used to learn an IPP solver to alleviate the computational burden of sampling-based IPP and further improve performance (Wei and Zheng, 2020; Cao et al., 2023; Vashisth et al., 2024). RL-based IPP pipelines typically consist of three components: (a) constructing a representation of the search space, (b) modeling environmental phenomena, and (c) training an RL agent to select paths that maximize information gain. For example, CAtNIPP (Cao et al., 2023) generates a probabilistic roadmap (PRM) (Geraerts and Overmars, 2004) to represent a continuous 2D search domain, reducing the complexity of the search space. (a) At the start of

each episode, the PRM generates a graph $G = (V, E)$ with nodes V and edges E, where each node is connected to k neighbors, and the agent is initialized at a random node. At each step, the agent moves to a neighboring node and observes the environmental phenomenon at its new location. (b) To model this phenomenon, Gaussian Process (GP) regression (Seeger, 2004; Wang, 2023) is used. Given observed locations X, the mean and covariance of a test location X^* are inferred as: $\mu = K(\mathcal{X}^*, \mathcal{X})[K(\mathcal{X}, \mathcal{X}) + \sigma_n^2 I]^{-1}(\mathcal{Y} - \mu(\mathcal{X})), P = K(\mathcal{X}^*, \mathcal{X}^*) - K(\mathcal{X}^*, \mathcal{X})[K(\mathcal{X}, \mathcal{X}) + \sigma_n^2 I]^{-1} K(\mathcal{X}, \mathcal{X})^T$, where $K(\cdot, \cdot)$ is the kernel function, σ_n^2 is the noise variance hyperparameter. The agent predicts the phenomenon via the GP and uses this prediction as input to the RL policy, while the GP is updated after each observation. The GP output is incorporated into the graph, forming a GP-augmented graph, which, together with the agent's current location, budget, and trajectory history, serves as input to the policy. (c) The RL agent then selects the next node. To train the agent to reduce GP uncertainty, the reward is defined as the normalized information gain, $r_t = (\mathrm{Tr}(P^{t-1}) - \mathrm{Tr}(P^t))/\mathrm{Tr}(P^{t-1})$, with an additional penalty $r_d = -\alpha \cdot \mathrm{Tr}(P^d)$ at the end of each episode. CAtNIPP uses PPO (Schulman et al., 2017) for training. As another example, Vashisth et al. (2024) proposes an RL-based IPP algorithm for 3D environments with unknown obstacles. To avoid collisions, they introduce a dynamically constructed graph representing the robot's local region, which is used as input to the RL agent. Additionally, they propose a new reward function to balance exploration and exploitation. Apart from these components, this 3D RL-based IPP algorithm follows the CAtNIPP framework.

These RL-based IPP methods achieve strong performance and fast planning at test time, but they require online interaction to collect data, which can be costly, risky, and time-consuming in physical systems. We instead formulate IPP as an offline RL problem, learning a policy from pre-collected trajectories to optimize long-horizon information gain rather than simply imitate logged actions. To the best of our knowledge, this is the first offline RL-based IPP solver.

3. Our approach

We propose `OffRIPP`, an offline RL-based informative path planning algorithm that trains an IPP solver using existing datasets without requiring interaction with the environment. By avoiding costly and risky real-time rollouts, `OffRIPP` is applicable to real-world scenarios. Moreover, `OffRIPP` can be integrated with any RL-based IPP algorithm, including CAtNIPP (Cao et al., 2023) and the 3D RL-based IPP method (Vashisth et al., 2024).

3.1. Problem Formulation

Dataset. We assume access to a dataset $\mathcal{D}$ generated by arbitrary algorithms, consisting of D episodes, where each episode contains sequences of states, actions, and rewards. Each state includes a graph representation in which each node contains a physical location and the agent's predicted environmental phenomenon, as well as the planning state, e.g., the current agent position and the remaining budget. Actions correspond to selecting the next node, and rewards correspond to reductions in GP uncertainty.

Using this dataset, our goal is to train an RL agent that plans a trajectory maximizing information gain under a budget constraint. Importantly, because the dataset already contains PRMs and GP-based environmental predictions, `OffRIPP` does not need to generate these components during training; instead, it directly leverages the given dataset. Once training is complete, the learned policy is deployed in the test environment for evaluation.

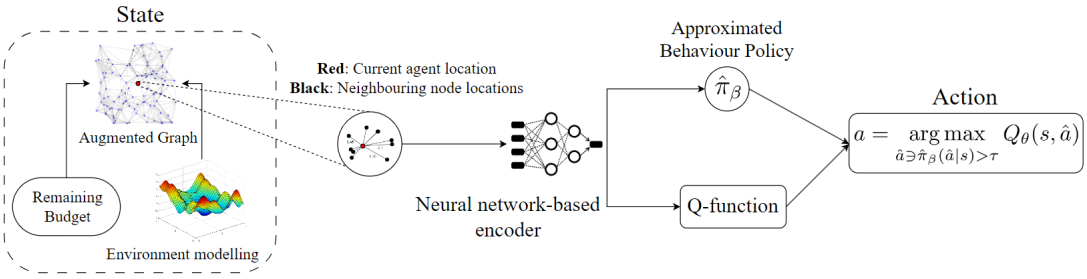


Figure 2: The architecture of `OffRIPP`: A graph augmented by environment modeling (the output of GP) and the remaining budget are used as input. The approximated behavior policy and Q-function are used to determine an action.

3.2. Offline RL-based Informative path planning

Naively training standard online RL algorithms such as PPO (Schulman et al., 2017) on offline datasets often leads to inaccurate value estimation and unstable learning, as discussed in Sec. 2.1. As we will discuss in Sec. 4.3, CAtNIPP, which relies on PPO, fails to learn from the offline dataset. To address this instability, we adopt the key idea of batch-constrained Q-learning (BCQ) (Fujimoto et al., 2019), which restricts the policy during training to select actions that remain close to those observed in the dataset. Concretely, a separate model is introduced to generate actions similar to those in the batch, and these actions are then used to construct the policy together with the Q-function. To this end, `OffRIPP` consists of two components: an RL agent and a behavior policy approximator. To improve sample efficiency, both components share the same neural network backbone except for the final layer. We now describe the details of each component in the following subsections.

3.2.1. BEHAVIOR POLICY APPROXIMATION

We build an approximation of the behavior policy, $\hat{\pi}_{\theta_\beta}$, to ensure that the learning policy avoids selecting actions that are not supported by the dataset. Since we only have access to the dataset generated by the behavior policy, and not the behavior policy itself, we utilize imitation learning, which minimizes the negative log-likelihood function. The loss function for the behavior policy approximator, parameterized by θ_β, is written as:

$$\mathcal{L}(\theta_\beta) = -\mathbb{E}_{(s,a)\sim\mathcal{D}} \left[\log \hat{\pi}_{\theta_\beta}(a|s) \right], \tag{3}$$

where (s, a) is the state-action pair in the dataset. We pretrain the behavior policy approximator before policy learning.

3.2.2. POLICY LEARNING

We build a policy based on the Q-function $Q_{\theta_Q}(s, a)$ and train it using the TD update. Instead of using a greedy policy as in Q-learning, following BCQ, we construct the policy using the behavior policy approximator as:

$$\pi(s) = \underset{\hat{a} \ni \hat{\pi}_\beta(\hat{a}|s) > \tau}{\arg\max} \; Q_\theta(s, \hat{a}), \tag{4}$$

where τ is the threshold that defines how much the learning policy deviates from the behavior policy. In other words, we restrict the policy to generate actions that the behavior policy is likely to generate

with a probability above the threshold τ. Note that $\tau = 0$ corresponds to fully imitating the behavior policy, while $\tau = 1$ corresponds to following the greedy policy. The rationale behind this is to avoid using unseen state-action pairs, which can lead to extrapolation errors. Based on this policy, we train the Q-function using TD update and the corresponding loss function is written as

$$\mathcal{L}(\theta_Q) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\Big[\Big(r + \gamma \max_{\hat{a}\ni\hat{\pi}_\beta(\hat{a}|s')>\tau} Q_{\bar{\theta}_Q}(s', \hat{a}) - Q_{\theta_Q}(s, a)\Big)^2\Big],$$

where $\bar{\theta}_Q$ represents the parameters of the target Q-function, which is a delayed update of the Q-function, providing stable targets to reduce instability during training (Mnih et al., 2015).

3.2.3. NETWORK ARCHITECTURE.

The RL policy and the behavior policy approximator share the entire network except for the final layer. The shared network architecture follows the design of either CAtNIPP (Cao et al., 2023) or the RL-based IPP method (Vashisth et al., 2024), and any existing RL-based IPP architecture can be integrated. Here, we describe an example based on CAtNIPP. The shared network takes an augmented graph as input, where each node contains its position and the GP mean and variance, and generates a representation that captures spatial relationships between nodes. This augmented graph is processed with positional encoding. The planning state, which includes the remaining budget and executed trajectory, is then combined with the augmented graph to form context-aware node embeddings. These embeddings are passed through an LSTM block, and the resulting features are processed by a network employing cross-attention between the current node and its neighboring nodes, producing a spatially aware representation of the current state.

On top of the shared network, two separate heads are used for the behavior policy and the Q-function. For the behavior policy, a final attention layer uses the neighboring features to extract the policy distribution. A binary mask M is applied to eliminate nodes that violate the budget constraint. For the Q-function, a simple MLP is used. The final action is obtained using Eq. 4, which combines both the approximated behavior policy and the Q-function. Our architecture is illustrated in Fig. 2.

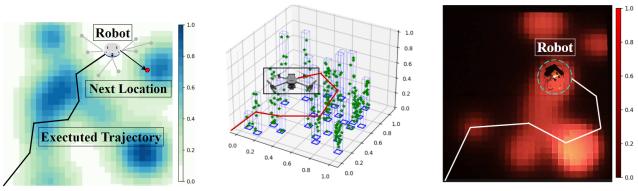
4. EXPERIMENTS

In this section, we validate the effectiveness of our proposed approach, `OffRIPP`, in two different simulated environments: a 2D light-intensity IPP task and a 3D fruit identification task, as well as in a real-world robotic environment. Since our work is the first to apply offline RL to IPP, we generate diverse datasets with varying performance levels, comprising both optimal and sub-optimal data, using existing RL-based IPP algorithms and a traditional IPP solver for evaluation. We believe the generated dataset will also make a valuable contribution to the research community.

4.1. Experiment Setup

4.1.1. LIGHT-INTENSITY ENVIRONMENT

Adopted from Cao et al. (2023), the goal in this environment is to estimate the light-intensity field across a 2D map. The light intensity is generated by sampling 8 to 12 random 2-dimensional Gaussian distributions within the unit square $[0, 1]^2$. The resulting field is illustrated in Fig. 3 (a). At the start of each episode, the agent's initial belief is set to a uniform distribution.



(a) Light-intensity (b) 3D Fruit Identification (c) Real-world Experiment

Figure 3: Three experiment settings: (a) Heatmap representing light intensity. (b) Green stars and blue structures represent the target fruits and tree. (c) Real-world experiment featuring a robot (dotted circle) with the intensity map projected onto the arena.

During dataset collection, both the agent's start and destination positions are randomly generated within the same unit square. The environment is discretized using a PRM with a fixed number of nodes, 400, and the number of neighboring nodes is set to 20. The agent collects measurements every 0.2 units of distance traveled, with additive sensor noise incorporated during measurement collection to evaluate robustness. While the budget is randomized between 6 and 8 during dataset collection, it is evaluated at fixed budgets of $6, 8$, and 10 during testing. The maximum episode length is fixed at 256 time steps. In this environment, `OffRIPP` is instantiated on top of CAtNIPP.

Dataset: We generate the following datasets using both CAtNIPP and a non-learning adaptive sampling method, with each dataset consisting of 18,500 trajectories.

• *Expert Dataset*: This dataset is collected by deploying the CAtNIPP greedy variant's best-performing model, which has been trained for over 50,000 episodes.

• *Medium Dataset*: This dataset is collected by deploying partially trained CAtNIPP greedy variant models, specifically those chosen between 256 and 512 episodes of training. This dataset consists of suboptimal trajectories.

• *Greedy Planning Dataset*: This dataset is collected using a simple heuristic adaptive sampling approach, where the agent selects its next node from among its neighbors based on the highest entropy of the GP. We refer to this method as greedy planning. Note that this method is non-learning-based and does not require training.

4.1.2. 3D FRUIT IDENTIFICATION

We use the 3D fruit identification environment introduced in Vashisth et al. (2024). The goal is to estimate the fruit distribution, illustrated in Fig. 3 (b). The environment is represented by a $50 \times 50 \times 50$ voxel occupancy grid, which is initially unknown and updated based on sensor observations. Trees are randomly placed, and fruits are attached to the trees at random positions. The agent observes part of the field of view, including free space, observed fruits, and trees. The action space consists of four discretized yaw angles. The reward function is based on the reduction in utility uncertainty of the GP and the number of observed targets. In this environment, `OffRIPP` is instantiated on top of the RL-based IPP method proposed in Vashisth et al. (2024).

Dataset: We generate the following datasets using an RL-based IPP solver and a traditional approach, with each dataset consisting of 18,500 trajectories.

• *Expert Dataset*: This dataset is collected by deploying a policy trained for approximately 10,000 interactions with the environment. The policy achieves high performance.

• *Medium Dataset*: This dataset is gathered using a partially trained policy, specifically models trained between 256 and 512 episodes. These policies are suboptimal.

• *Greedy Planning Dataset*: This dataset is collected using a simple heuristic adaptive sampling approach, similar to the 2D environment. The agent selects its next node based on the highest entropy in the GP model, without requiring any learning or training.

Table 1: Performance comparison in terms of the trace of the covariance matrix in the light-intensity environment. A lower value indicates reduced uncertainty, indicating better performance. The behavior policy is not a baseline.

Model	Budget 6	Budget 8	Budget 10
Expert Dataset			
Behavior Policy (Dataset)	18.48 ± 7.76	6.96 ± 2.68	3.73 ± 1.57
CAtNIPP (Offline Trained)	137.5 ± 167.2	125.8 ± 203.7	110.3 ± 245.1
BC	30.84 ± 14.02	9.93 ± 5.03	7.02 ± 3.06
OffRIPP	**23.28** ± 5.80	**7.83** ± 2.87	**3.96** ± 1.41
Medium Dataset			
Behavior Policy (Dataset)	32.84 ± 7.98	18.20 ± 5.69	9.78 ± 2.25
CAtNIPP (Offline Trained)	480.8 ± 199.2	557.4 ± 207.5	628.6 ± 182.7
BC	48.08 ± 33.02	23.59 ± 19.07	11.36 ± 9.47
OffRIPP	**34.47** ± 27.41	**17.91** ± 10.54	**8.10** ± 4.42
Greedy Planning Dataset			
Behavior Policy (Dataset)	73.21 ± 99.80	65.00 ± 102.84	60.46 ± 104.41
CAtNIPP (Offline Trained)	433.1 ± 169.1	517.8 ± 195.7	602.9 ± 176.5
BC	42.25 ± 27.91	**16.39** ± 10.27	10.04 ± 5.86
OffRIPP	**39.16** ± 22.42	16.56 ± 12.82	**7.61** ± 5.75
Non-learning-based IPP solvers			
Greedy Planning	73.21 ± 99.80	65.00 ± 102.84	60.46 ± 104.41
RAOr	49.47 ± 20.29	19.87 ± 7.71	12.54 ± 5.13

Table 2: Performance comparison in terms of the fruit detection rate in the 3D fruit identification environment. A higher value indicates better performance. Note that the behavior policy is not a baseline.

Model	Budget 6	Budget 8	Budget 10
Expert Dataset			
Behavior Policy (Dataset)	46.17 ± 6.94	54.61 ± 6.41	59.12 ± 7.82
3D RL-based IPP (Offline Trained)	26.85 ± 9.86	32.55 ± 10.75	36.53 ± 12.40
BC	45.98 ± 7.86	53.87 ± 7.12	59.69 ± 7.51
OffRIPP	**46.60** ± 8.87	**55.57** ± 6.34	**61.48** ± 6.88
Medium Dataset			
Behavior Policy (Dataset)	27.32 ± 13.31	32.92 ± 15.73	38.55 ± 18.09
3D RL-based IPP (Offline Trained)	24.31 ± 10.96	30.45 ± 8.80	36.21 ± 12.23
BC	16.00 ± 6.41	19.86 ± 6.79	20.31 ± 7.82
OffRIPP	**29.27** ± 11.08	**32.33** ± 10.20	**36.37** ± 11.25
Greedy Planning Dataset			
Behavior Policy (Dataset)	17.29 ± 13.09	17.35 ± 14.10	31.02 ± 13.82
3D RL-based IPP (Offline Trained)	24.40 ± 10.63	29.43 ± 11.02	29.41 ± 10.26
BC	7.64 ± 7.44	11.39 ± 10.48	8.72 ± 8.34
OffRIPP	**25.28** ± 11.31	**29.71** ± 9.20	**36.26** ± 10.82

4.2. Training Details

To train OffRIPP, we use the Adam optimizer (Kingma, 2014), a batch size of 256, and an update frequency of 100 for the target network. For the threshold τ in Eq. 4, we tune τ in the range $(0, 1)$. OffRIPP is trained for one epoch over the dataset, requiring on average 3 hours in the light-intensity environment and 18 hours in the 3D fruit identification environment on a workstation equipped with an Intel Xeon Gold 5218 CPU and three NVIDIA RTX 6000 GPUs.

4.3. Experimental Results

4.3.1. LIGHT-INTENSITY ENVIRONMENT

For the evaluation of OffRIPP, we consider four baselines: (i) *Greedy Planning*, where the agent selects its next node based on the highest GP entropy among neighbors; (ii) Randomized Anytime Orienteering (RAOr) (Arora and Scherer, 2017), a sampling-based heuristic. Note that *Greedy Planning* and RAOr do not require training, so we report their performances regardless of the dataset; (iii) Behavior Cloning (BC), which imitates the behavioral policy. We use the CAtNIPP architecture for BC and train it via negative log-likelihood; and (iv) CAtNIPP (Cao et al., 2023) trained offline, where the agent is trained with PPO on the dataset. In addition, we report the dataset performance itself to assess whether the learned model can surpass the model that generated the data. Note that the behavior policy is trained online, making direct comparisons with OffRIPP unfair.

We evaluate all methods using the three dataset types from Sec. 4.1 with budget values of 6, 8, and 10. The metric is the average trace of the covariance matrix over 50 environment instances, which is the optimization objective in RL and also used by Cao et al. (2023). Results are shown

in Table 1. OffRIPP consistently achieves the lowest covariance trace across all budget settings. Notably, CAtNIPP performs poorly when trained offline, illustrating that naive online RL algorithms are unstable in the offline setting. OffRIPP, however, performs well and often exceeds the dataset performance. Furthermore, despite sensor noise during data collection, OffRIPP remains robust, maintaining accurate predictions and effective path planning, highlighting its applicability to real-world deployment. Finally, OffRIPP trained on any dataset outperforms both non-learning-based baselines Greedy Planning and RAOr.

4.3.2. 3D FRUIT IDENTIFICATION ENVIRONMENT

For evaluation, we consider two baselines: (i) BC and (ii) the 3D RL-based IPP solver proposed in Vashisth et al. (2024), as detailed in Sec. 2.2. These models are assessed across the three dataset types under budget constraints of 6, 8, and 10, as described in Sec. 4.1. The evaluation metric is the average fruit detection rate across 50 environment instances, following the same comparative analysis procedure used in Vashisth et al. (2024). This metric captures the accuracy and efficiency of the models in detecting fruits within a limited budget, offering a reliable means of comparison. As shown in Table 2, OffRIPP consistently achieves higher fruit detection rates across all budget levels, outperforming both BC and the 3D RL-based IPP solver. The 3D RL-based IPP solver, designed for online RL, performs suboptimally in the offline training setting, illustrating its instability under distribution shift. In contrast, OffRIPP not only outperforms these baselines but also exceeds the dataset's performance in many cases. Notably, on the Greedy Planning dataset generated by a non-learning method, we observe substantial improvement.

4.3.3. PERFORMANCE WITH RESPECT TO THE DATASET

The performance of OffRIPP is influenced by both the quality and quantity of the dataset, since OffRIPP depends on the given dataset without generating new data from the current policy. From Table 1 and Table 2, we observe that performance improves as the behavior policy quality increases. Fig. 4 shows the performance of OffRIPP and RAOr as the dataset size varies. Performance improves as the number of training episodes increases, with convergence beginning around $18k$ episodes, suggesting that roughly $18k$ episodes are needed for near-optimal performance. Furthermore, OffRIPP requires only 500 episodes to outperform RAOr, a non-learning-based IPP solver.

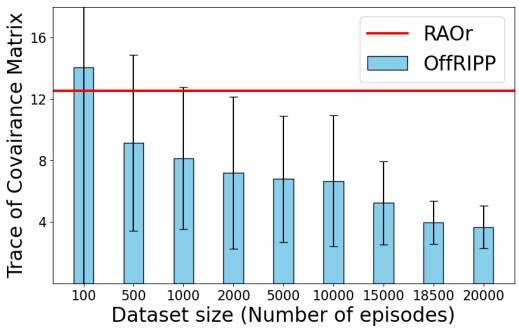


Figure 4: Performance of OffRIPP across sizes of the expert dataset in the light-intensity environment (budget = 10). Lower values indicate better performance. RAOr does not require a dataset because it is non-learning-based.

4.4. Analysis

4.4.1. TOTAL PLANNING TIME

One of the advantages of OffRIPP is fast planning during execution. We report the total planning time for executing one trajectory for OffRIPP, BC, and RAOr in the light-intensity environment.

OffRIPP requires 0.64 seconds, slightly more time compared to BC, which requires 0.62 seconds, due to the additional Q-value estimation module. However, it is significantly faster than RAOr, which requires 6.11 seconds, a strong non-learning-based IPP baseline. Therefore, OffRIPP outperforms the non-learning method in both performance and planning time.

4.4.2. OFFLINE RL ALGORITHMS

As described in Sec. 3.2, we leverage a key idea from BCQ, which constrains the policy by approximating the behavior policy. This approach, known as behavior-constrained offline RL, has been effective for IPP so far. To further investigate offline RL for IPP, we also employ another offline RL algorithm, CQL (Kumar et al., 2020), described in Sec. 2.1. Unlike BCQ, CQL uses a conservative value estimation approach by constraining the critic to assign lower values to unseen data rather than directly constraining the policy. Although BCQ demonstrates strong performance in our method, we found that CQL underperforms for IPP due to the necessity of action constraints for stable learning in this problem. The performance of CQL in the light-intensity environment with the expert dataset is 474.7 ± 113.6, which is considerably worse than our approach. This suggests that explicit action support constraints are more crucial than conservative critics for stable offline learning in IPP. Further exploration of other offline RL algorithms is left as future work.

4.5. Experimental Validation on Real Robot

We validate OffRIPP on a real-robot system in a light-intensity environment. Specifically, we project the light-intensity environment onto a physical 1.5×1.5 m^2 arena, as shown in Fig. 3 (c), maintaining consistency in configuration between simulation and physical deployment. The Khepera-IV robot, equipped with a Raspberry Pi 3 and a camera module, is then deployed. In this experiment, we train OffRIPP offline using the expert dataset and deploy the resulting policy on the robot. During deployment, the robot collects real-time image observations and executes actions based on those observations. Despite real-world challenges such as sensor noise and environmental uncertainties, OffRIPP demonstrates robust performance, successfully navigating the environment and making reliable decisions. These results further validate the adaptability of OffRIPP, as it generalizes from simulation to reality without requiring retraining or extensive modifications. Notably, we achieve a covariance trace of 4.83 in the real-world deployment, highlighting its effectiveness in maintaining informative exploration under real-world conditions. The full video can be accessed here.

5. CONCLUSION

In this paper, we presented an offline RL-based IPP framework that trains an RL agent to find paths that maximize information gain without environment interactions during training. Building on an IPP architecture with graph representations, GP-based environment modeling, and attention mechanisms, we approximate the behavior policy used to generate the dataset and train a Q-function. Using this approximation, we derive a policy that selects high-value actions supported by the dataset. We show that OffRIPP outperforms baseline methods, including traditional IPP solvers, in both performance and planning time across two different environments. Furthermore, we validate the effectiveness of OffRIPP on a real-robot system. Future directions include extending this framework toward multi-agent IPP policies learned from offline datasets.

Acknowledgments

This work has been supported by NSF and USDA-NIFA under AI Institute for Resilient Agriculture, Award No. 2021-67021-35329 and Department of Agriculture Award Number 2023-67021-39073. The authors would like to thank the RISS program for providing the opportunity to conduct this research. A special thanks to Mrs. Rachel Burcin, Dr. John M. Dolan, Ms. Morgan Grimm, and the entire CMU Robotics Institute community for their support. The authors also gratefully acknowledge Dr. Simon Stepputtis for his assistance in conducting real-world experiments.

References

Sankalp Arora and Sebastian Scherer. Randomized algorithm for informative path planning with budget constraints. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4997–5004. IEEE, 2017.

Yuhong Cao, Yizhuo Wang, Apoorva Vashisth, Haolin Fan, and Guillaume Adrien Sartoretti. Catnipp: Context-aware attention-based network for informative path planning. In *Conference on Robot Learning*, pages 1928–1937. PMLR, 2023.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

Roland Geraerts and Mark H Overmars. A comparative study of probabilistic roadmap planners. In *Algorithmic foundations of robotics V*, pages 43–57. Springer, 2004.

Gregory Hitz, Enric Galceran, Marie-Ève Garneau, François Pomerleau, and Roland Siegwart. Adaptive continuous-space informative path planning for online environmental monitoring. *Journal of Field Robotics*, 34(8):1427–1449, 2017.

Austin Jones, Mac Schwager, and Calin Belta. A receding horizon algorithm for informative path planning with temporal logic constraints. In *2013 IEEE International Conference on Robotics and Automation*, pages 5019–5024. IEEE, 2013.

Siva Kailas, Wenhao Luo, and Katia Sycara. Multi-robot adaptive sampling for supervised spatiotemporal forecasting. In *EPIA Conference on Artificial Intelligence*, pages 349–361. Springer, 2023.

Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011.

Jeonghye Kim, Suyoung Lee, Woojun Kim, and Youngchul Sung. Decision convformer: Local filtering in metaformer is sufficient for decision making. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=af2c8EaKl8.

Jeonghye Kim, Suyoung Lee, Woojun Kim, and Youngchul Sung. Value-aided conditional supervised learning for offline rl. *arXiv e-prints*, pages arXiv–2402, 2024b.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Ajith Anil Meera, Marija Popović, Alexander Millane, and Roland Siegwart. Obstacle-aware adaptive informative path planning for uav-based target search. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 718–724. IEEE, 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Marija Popović, Teresa Vidal-Calleja, Gregory Hitz, Jen Jen Chung, Inkyu Sa, Roland Siegwart, and Juan Nieto. An informative path planning framework for uav-based terrain monitoring. *Autonomous Robots*, 44(6):889–911, 2020.

Julius Rückin, Federico Magistri, Cyrill Stachniss, and Marija Popović. An informative path planning framework for active learning in uav-based semantic mapping. *IEEE Transactions on Robotics*, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Matthias Seeger. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

Gerald Tesauro et al. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

Apoorva Vashisth, Julius Ruckin, Federico Magistri, Cyrill Stachniss, and Marija Popovic. Deep reinforcement learning with dynamic graphs for adaptive informative path planning. *IEEE Robotics and Automation Letters*, 2024.

Jie Wang. An intuitive tutorial to gaussian processes regression. *Computing in Science & Engineering*, 2023.

Yongyong Wei and Rong Zheng. Informative path planning for mobile sensing with reinforcement learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 864–873. IEEE, 2020.

Soo-Hyun Yoo, Andrew Stuntz, Yawei Zhang, Robert Rothschild, Geoffrey A Hollinger, and Ryan N Smith. Experimental analysis of receding horizon planning algorithms for marine monitoring. In *Field and Service Robotics: Results of the 10th International Conference*, pages 31–44. Springer, 2016.