DyPNIPP: Predicting Environment Dynamics for RL-based Robust Informative Path Planning

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) aims to find a path that maximizes information gain while adhering to planning constraints so that robots can learn an accurate belief of the quantity of interest, applicable to various real-world robotic applications such as environment monitoring. Traditional IPP methods typically require high computation time during execution, giving rise to reinforcement learning (RL) based IPP methods. However, existing RL-based approaches largely focus on static spatial environments and do not consider spatio-temporal environments where the underlying dynamics evolve over time. In this paper, we propose `DyPNIPP`, a robust RL-based IPP framework, designed to operate effectively across spatio-temporal environments with varying dynamics. To achieve this, `DyPNIPP` incorporates domain randomization to train the agent across diverse environments and introduces a dynamics prediction model to capture and adapt the agent actions to specific environment dynamics. Our extensive experiments in a wildfire environment demonstrate that `DyPNIPP` outperforms existing RL-based IPP algorithms by significantly improving robustness and performing across diverse environment conditions.

Keywords: Robustness, Reinforcement learning, Informative path planning, and Spatiotemporal dynamics

1. Introduction

Informative path planning (IPP) has been actively studied for robotics deployments involving information acquisition, such as the autonomous exploration of unknown areas (Liang et al., 2023), environment monitoring (Hitz et al., 2017), and target tracking (Wang et al., 2023). IPP aims to find a path for autonomous robots that maximizes the acquisition of interests, e.g., the intensity of fire in fire monitoring, while adhering to resource constraints. Conventional IPP methods typically involve sampling-based path planning using a graph for spatial environments (Hitz et al., 2017; Karaman and Frazzoli, 2011; Arora and Scherer, 2017). Recently, IPP solvers for spatio-temporal environments, where the interest changes over time, have also been proposed (Jakkala and Akella, 2023; Kailas et al., 2023). Despite their effectiveness, these methods require heavy computation time to determine the path, limiting their applicability for real-world deployment.

With the recent success of RL in various domains, RL-based IPP has been studied, demonstrating both superior performance and reduced computation time (Cao et al., 2023; Rückin et al., 2022; Vashisth et al., 2024). However, existing works do not consider spatio-temporal environments, where learning faces an inherent RL challenge: a lack of robustness against variations in environment dynamics (Popovic et al., 2024). That is, the agent performs optimally only in the environment it was trained on, but not when the dynamics differ. Even when trained over multiple dynamics, policies tend to be over-regularized and still exhibit suboptimal performance across variations (Tiboni et al., 2023). To illustrate this robustness issue, consider a wildfire domain where the interest is the fire that spreads over time and the spread rate is determined by several characteristics, including fuel and wind velocity. Fig. 1 shows two environments with different fuel distributions, denoted by F_c: F_c=1 (top, slow spread) and F_c=10 (bottom, fast spread). Here, the agent trained on the F_c=1 environment has suboptimal performance in the F_c=10 environment, and vice versa. Table 2 shows the corresponding cumulative RMSE between predicted and ground-truth environments after each step. Notably, CAtNIPP[1] (Cao et al., 2023) trained on F_c=10 performs worse than that trained on F_c=1 in the F_c=1 environment, implying that the existing RL-based algorithm lacks robustness against variations in environment dynamics.

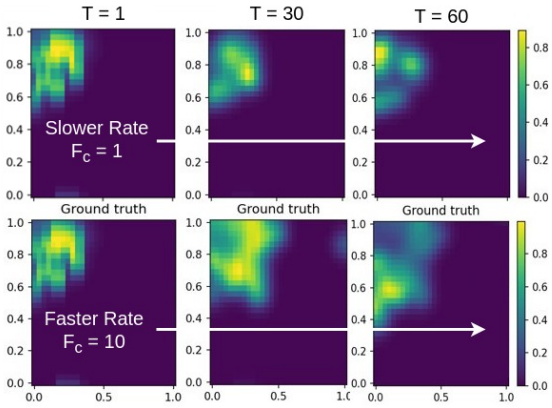

Figure 1: Wildfire environments with different dynamics: $F_c = 1$ (top) and $F_c = 10$ (bottom).

Model	Fuel coefficient	
	$F_c = 1$	$F_c = 10$
CAtNIPP (trained on F_c=1)	13.7 ± 2.6	25.0 ± 5.3
CAtNIPP (trained on F_c=10)	16.2 ± 3.6	20.9 ± 3.0
CAtNIPP + DR	14.9 ± 2.4	24.0 ± 5.6
DyPNIPP (Ours)	$\mathbf{10.3 \pm 1.6}$	$\mathbf{14.7 \pm 2.9}$

Figure 2: RMSE results in the wildfire environment shown in Fig 1.

In this paper, we propose a robust RL-based IPP framework named DyPNIPP, capable of operating effectively across environments with varying dynamics. First, DyPNIPP adopts domain randomization (DR), which randomizes environment characteristics, e.g., fuel distribution in the wildfire domain, to train an agent across a diverse set of environments. However, DR alone is insufficient, as shown in Table 1, since the agent tends to learn an averaged policy and cannot infer environment-specific dynamics (Tiboni et al., 2023). We thus introduce a dynamics prediction model that captures environment dynamics *implicitly*, allowing the RL policy to recognize the current dynamics and adapt decisions accordingly. We evaluate DyPNIPP in wildfire simulations with multiple sources of variation, including fuel coefficient and number of fires, as well as in an air temperature prediction environment. We show that DyPNIPP outperforms both learning-based and non-learning IPP algorithms across varying dynamics, demonstrating robustness.

Our main contributions are: (i) To the best of our knowledge, this is the first work addressing robustness in RL-based informative path planning under spatio-temporal environments. (ii) We consider key robustness factors in wildfire scenarios, including fuel coefficients and the number of fires. (iii) We demonstrate the effectiveness of DyPNIPP across diverse conditions and provide analyses, along with a real-robot experiment validating practical deployment.

1. This is a prior work on RL-based IPP. This will be discussed later.

2. Background and Related Works

2.1. Reinforcement Learning

Reinforcement learning (RL) trains an agent through interaction with an environment, typically formulated as a Markov decision process (MDP). At each time step t, the agent selects an action a_t from a policy π based on the given state s_t. The environment then yields a reward $r_t = r(s_t, a_t)$ and the next state based on the transition probability $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$. The goal is to learn a policy that maximizes the expected discounted return $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$. One representative algorithm is proximal policy optimization (PPO) (Schulman et al., 2017). PPO consists of an actor and a value function (critic), which estimates expected returns. The actor collects experience, and the value function estimates advantages, indicating how much better or worse an action is compared to the current policy expectation. The actor is then updated to maximize these advantages. PPO uses a clipping method to limit how much the new policy can deviate from the previous one, ensuring stable and efficient learning.

2.2. Robustness

Robustness is a key requirement for deploying machine learning models in real-world settings. Robustness can take different meanings depending on context, such as resistance to adversarial attacks (Pattanaik et al., 2017; Zhang et al., 2020) or robustness to distribution shifts (Chae et al., 2022; Lee et al., 2020). In this paper, we focus on robustness to variations in environment dynamics within the context of RL. One line of work introduces robust MDPs, which add an uncertainty transition set and optimize worst-case expected return (Derman et al., 2018; Mankowitz et al., 2018, 2019). Another line is domain randomization (Tan et al., 2018; Peng et al., 2018; Slaoui et al., 2019; Zhao et al., 2020), which randomizes environment dynamics to improve policy generalization. For instance, Peng et al. (2018) leverages domain randomization to develop an RL policy that is robust to variations in environment dynamics, thereby achieving robust sim-to-real transfer. The other direction is modeling dynamics explicitly via meta-learning or supervised learning (Nagabandi et al., 2018; Lee et al., 2020), where context-aware dynamics prediction enables adaptation to dynamic changes. These approaches have been applied to various robotics domains, but none have been considered for informative path planning.

2.3. Informative Path Planning

The objective of informative path planning is to find an optimal trajectory ψ^* over the set of feasible trajectories Ψ that maximizes an information-theoretic objective:

$$\psi^* = \arg\max_{\psi \in \Psi} I(\psi), s.t.\ C(\psi) \leq B, \tag{1}$$

where $I : \psi \to \mathbb{R}^+$ represents the information gain from measurements obtained along the trajectory ψ, $C : \psi \to \mathbb{R}^+$ maps a trajectory ψ to its execution cost, and B denotes the robot's budget limit, e.g., path length, time, or energy. In this work, we consider path-length constraints and define the information gain as $I(\psi) = \text{Tr}(P^-) - \text{Tr}(P^+)$, where $\text{Tr}(\cdot)$ denotes the trace of a matrix, and P^- and P^+ represent the prior and posterior covariances, respectively, obtained before and after taking measurements of the underlying environmental phenomenon along the trajectory ψ, following prior works (Popović et al., 2020; Cao et al., 2023). The information gain is computed from sensor

measurements taken at fixed intervals along the trajectory ψ. Measurements are collected each time the robot travels a fixed distance, and as a result, the total number is determined by the path-length budget B.

Optimizing Eq. 1, i.e., solving IPP, is known to be NP-hard. Consequently, computationally tractable approximate methods, such as sampling techniques that explore a complex space by generating random samples of possible solutions, have been proposed (Hitz et al., 2017; Karaman and Frazzoli, 2011; Arora and Scherer, 2017; Jones et al., 2013; Yoo et al., 2016). However, these sampling-based methods still require heavy computation at test time, which restricts their use in real-world applications.

RL-based IPP: To address the aforementioned problem, RL has been utilized to learn an IPP (Cao et al., 2024, 2023; Rückin et al., 2022; Vashisth et al., 2024; Gadipudi et al., 2024). Most RL-based IPP algorithms are composed of three modules: (a) creating a representation of the entire search map, (b) modeling environmental phenomena, and (c) training an RL agent. One representative example is CAtNIPP (Cao et al., 2023). CAtNIPP trains an RL policy for IPP in static, time-invariant 2D environments. Before the training starts, CAtNIPP uses a *probabilistic roadmap* (PRM) (Kavraki et al., 1996) that covers the continuous search domain to decrease the complexity of the search space. PRM generates a route graph, $G = (V, E)$, where V and E are sets of nodes and edges, respectively, and each node has k neighbor nodes. Here, the agent is initialized on a randomly chosen node. Next, CAtNIPP leverages GP regression (Seeger, 2004) to model the spatial phenomena of the search space, and the output of the GP regression is referred to as the belief of the phenomena. At each time step, the agent observes a measurement at the current node and then uses it to update the belief $\mathcal{GP}(\mu, P)$. Given a set of n' locations $\mathcal{X}^* \subset \mathcal{E}$ at which interest needs to be inferred, a set of n observed locations $\mathcal{X} \subset \mathcal{E}$ and the corresponding measurements set $\mathcal{Y}$, the mean and covariance of the GP is inferred as follows: $\mu = \mu(\mathcal{X}^* + K(\mathcal{X}^*, \mathcal{X})[K(\mathcal{X}, \mathcal{X}) + \sigma_n^2 I]^{-1}(\mathcal{Y} - \mu(\mathcal{X}))$, $P = K(\mathcal{X}^*, \mathcal{X}^*) - K(\mathcal{X}^*, \mathcal{X})[K(\mathcal{X}, \mathcal{X}) + \sigma_n^2 I]^{-1} \times K(\mathcal{X}^*, \mathcal{X})^T$, where $K(\cdot)$ is a pre-defined kernel function, σ_n^2 is a parameter representing the measurement noise, and I is a $n \times n$ identity matrix. In this paper, we consider Matérn $3/2$ kernel, following the prior works (Cao et al., 2023; Popović et al., 2020). Lastly, for training the RL policy, the graph augmented by the updated belief—where each node $v_i' = (v_i, \mu(v_i), P(v_i)) \in V$—along with the planning state, which includes the current location, the budget, and the executed trajectory so far, is used as input for the RL policy. Based on such information, the agent chooses one of the neighboring nodes to move to (action). The reward function is based on the previously described information gain, which represents the reduction in uncertainty of GP regression, and is written as $r(t) = (Tr(P^{t-1}) - Tr(P^t))/Tr(P^{t-1})$. Summarized above, CAtNIPP trains an RL policy that chooses the neighboring node to move to based on the updated belief-augmented graph to maximize the expected sum of the reduction in uncertainty of GP regression. Additionally, CAtNIPP constructs the RL policy with an attention-based encoder and an LSTM-based decoder module.

In addition to CAtNIPP, several RL-based IPP algorithms, including Vashisth et al. (2024) where a dynamic graph is proposed to ensure collision-free navigation in 3D environments, have been introduced. The prior works have been shown to be effective in static, time-invariant environments however, none have considered spatio-temporal environments or variations in environmental dynamics, where the testing environments differ from those on which the agent was trained. To the best of our knowledge, our work is the first to propose an RL-based IPP policy for spatio-temporal environments and rigorously address its robustness.

3. Methodology

3.1. Motivation

As described in Sec. 2.1, the standard RL framework assumes a fixed, stationary transition probability $p(s' \mid s, a)$ that captures the environment dynamics. However, the optimal policy learned under a specific transition probability may perform suboptimally when the dynamics differ. To formalize this, we consider a distribution over MDPs in which the transition probability $p(s' \mid s, a, c)$ is conditioned on environment characteristics c. These characteristics depend on the domain; for example, in the wildfire domain, c includes factors that affect fire dynamics, such as fuel distribution and wind velocity, which are unknown to the agent.

The existing RL-based IPP algorithms (Cao et al., 2023; Vashisth et al., 2024) are designed for static, time-invariant environments and do not explicitly account for variations in environment dynamics. As a result, an RL policy trained on specific environment characteristics tends to perform well only in the environment it was trained on, but degrades when deployed under different dynamics (Sec. 4.3.1). Since the agent can encounter a wide range of environments with different characteristics, it must be capable of handling such variations.

3.2. Proposed Method: `DyPNIPP`

We aim to train an RL policy for IPP that performs well across environments with varying dynamics, i.e., a policy that is robust to such variations. To this end, we present `DyPNIPP`: an environment **Dy**namics **P**rediction **N**etwork for **IPP**. The proposed framework comprises two components: (1) domain randomization (DR), and (2) a dynamics prediction model that predicts the belief of the next observation and enables the RL policy to capture environment dynamics. Note that `DyPNIPP` can be integrated with any RL-based IPP algorithm.

3.2.1. DOMAIN RANDOMIZATION

In order for the RL agent to encounter a diverse range of environment characteristics, we adopt domain randomization, which randomizes environment dynamics by sampling characteristics from a specified range. This can be viewed as marginalizing the environment dynamics over the characteristic distribution: $p(s'|s, a) = \mathbb{E}_c\left[p(s'|s, a, c)\right]$. Concretely, at the beginning of each episode, we sample a characteristic c from a prior distribution (e.g., $\mathrm{Uniform}[1, 10]$ for fuel coefficient in the wildfire domain) and train the RL policy in that environment. However, as we show in Sec. 4.3.1, domain randomization alone is insufficient: the learned policy becomes optimal only for the marginalized dynamics, $\mathbb{E}_c[p(s'|s, a, c)]$, and not for each individual environment with a specific characteristic.

3.2.2. DYNAMICS PREDICTION MODEL

To address the aforementioned limitation, we introduce a dynamics prediction model (DPM) capable of extracting features that represent environment dynamics. The DPM consists of an encoder composed of convolutional layers followed by an LSTM, and a decoder composed of dense layers and transposed convolutional layers, parameterized by ϕ_{enc} and ϕ_{dec}. The encoder computes a latent vector $z_t = f(b(o_t), h_t; \phi_{enc})$ from the belief of the current observation $b(o_t)$ and the LSTM hidden state h_t, while the decoder predicts the belief of the next observation $y_t = b(o_{t+1})$ as $g(z_t; \phi_{dec})$.

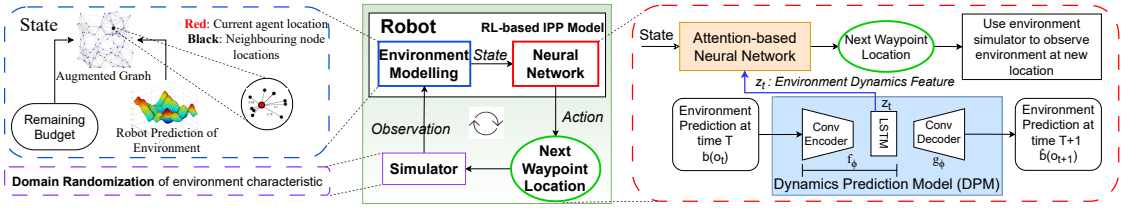

Figure 3: Overview of our approach: (Left) Environment modeling to build the input for the policy and domain randomization for the simulator. (Middle) The overall operation of the proposed IPP algorithm. (Right) Generating the next waypoint (action) using the RL policy and the proposed DPM (blue). DPM predicts the next environment state, and the environment dynamics feature is extracted from the hidden layer to serve as input to the RL policy.

The DPM is trained to minimize:

$$\mathcal{L}_{DPM}(\phi_{enc}, \phi_{dec}) = \mathbb{E}\left[\|y_t - g(z_t; \phi_{dec})\|^2\right] \tag{2}$$

By learning to predict the next belief, the latent representation z_t implicitly captures the underlying environment dynamics from observations. We use z_t as an environment-context feature to condition the RL policy. Combined with DR, the DPM enables the agent to become aware of environment-dependent dynamics, which in turn improves robustness across varying environments.

3.2.3. RL POLICY FOR DYNAMICS-AWARE IPP

Based on the two proposed components, we train an RL policy that selects the next waypoint. `DyPNIPP` can be combined with any RL-based IPP method; in this work, we adopt CAtNIPP (Cao et al., 2023). The key difference is that we augment the policy with the environment-context feature z_t from DPM, resulting in a policy represented as $\pi(a_t \mid s_t, z_t)$.

The policy network consists of an attention-based encoder and an attention and LSTM-based decoder. The encoder takes the belief-augmented graph and produces spatial embeddings for all nodes. In addition, a planning-state embedding is generated from the remaining budget, the executed trajectory, and the augmented graph. The environment-context feature z_t is fused with these embeddings and passed into the LSTM. The decoder then selects the most informative neighbor of the robot via a cross-attention mechanism. The attention weights reflect the relevance of each neighbor under the current planning state and effectively determine the action.

This architecture uses the budget, executed trajectory, spatial information, and environment-context feature, enabling the agent to generate actions that are both feasible under the robot's operational constraints and informative given the environment dynamics. Note that the LSTM in the RL policy focuses on capturing past information related to planning, while the LSTM in the DPM is explicitly trained to capture environment dynamics. Following prior work (Cao et al., 2023), we train the policy using PPO.

Summarizing the overall operation: (1) we initialize the environment via domain randomization, (2) the robot observes the underlying phenomenon and updates the GP, (3) the GP-augmented graph and remaining budget are provided to the network composed of the RL policy and DPM, (4) the DPM infers the environment-dynamics feature, and (5) the RL policy generates an action selecting the next waypoint. This process is repeated throughout training for both the RL policy and the DPM. The overall framework is illustrated in Fig. 3.

4. Experiments

4.1. Environment Setup

We evaluate DyPNIPP on two domains: a wildfire domain based on the Fire Area Simulator (FARSITE) (Finney, 1998) model, and an air temperature prediction domain.

For the wildfire domain, we use the FireCommander (Seraj et al., 2020) simulator to generate spatio-temporal fire propagation environments. The fire spread dynamics depend on environment characteristics including the fuel coefficient F_c, wind speed U_c, and wind azimuth θ_c. For robustness evaluation, we vary F_c to create environments with different fire propagation dynamics, while randomizing θ_c at the start of each episode. We also evaluate robustness under variation in the number of initial fire origins. Detailed equations governing fire propagation are provided in the Appendix A.

For the air temperature prediction domain, we use real-world historical data (Kalnay et al., 2018) (monthly records from 1948–2022), discretized at 2.5° latitude $\times$ 2.5° longitude. We sample a region for both training and testing. Unlike wildfire, where dynamics shift across episodes through known parameters, temperature evolves over time naturally with unknown and evolving dynamics. Through both domains, we evaluate whether DyPNIPP can handle dynamic variation arising from both known structured parameters and naturally varying phenomena.

4.2. Training Details

In the FireCommander simulator, both the fuel coefficient F_c and wind speed U_c positively influence the fire spread rate ($F_c, U_c > 0$). Since F_c is the dominant factor affecting fire propagation, we fix U_c to 5 and vary F_c to evaluate robustness. The wind azimuth θ_c is randomly sampled at the beginning of each episode.

We apply domain randomization by sampling $F_c \in [1, 10]$ for every episode. Given these sampled parameters, FireCommander generates the spatio-temporal wildfire environment. The field is normalized to the unit square $[0, 1]^2$, and the robot's initial belief is initialized as a uniform GP prior $\mathcal{GP}(0, P^0)$ with $P^0_{i,i} = 1$. The start and destination locations are randomly sampled from $[0, 1]^2$. We train the policy using a graph with 200 nodes, $k = 20$ neighbors, and a budget randomized in $[7, 9]$. A measurement is obtained each time the robot moves 0.2 units. We set the episode length to 256 time steps and use a batch size of 32. Adam is used with learning rate 10^{-4} decayed every 32 steps by a factor of 0.96. DyPNIPP trains the RL agent on top of CAtNIPP (Cao et al., 2023) using PPO (Schulman et al., 2017). Each episode performs 8 PPO iterations. Training is executed on a workstation with an AMD EPYC 7713 CPU and a single NVIDIA RTX 6000 Ada GPU, and convergence is achieved in approximately 2 hours.

4.3. Experimental Results

In this subsection, we examine (1) whether DyPNIPP enhances the robustness against environment variations such as the fuel coefficient F_c and number of fires, (2) the effectiveness of our DPM design, and (3) how the environment-context latent feature varies with respect to the dynamics.

4.3.1. PERFORMANCE COMPARISON OF VARIATION IN FUEL COEFFICIENT

We include three main baselines: First, we include three CAtNIPP policies trained on a fixed $F_c \in 1, 5, 10$. Second, we include CAtNIPP combined with domain randomization, where F_c is ran-

Model used	Environment Parameter (Fuel and Vegetation coefficient)								
	budget = 7			budget = 11			budget = 15		
	F_c=1	F_c=5	F_c=10	F_c=1	F_c=5	F_c=10	F_c=1	F_c=5	F_c=10
Sampling-based IPP (non-RL)	523.1 ± 592.7	551.9 ± 505.5	550.2 ± 469.7	949.4 ± 1161.2	932.0 ± 870.2	997.4 ± 916.6	1305.2± 1643.1	1372.7± 1289.7	1367.1± 1142.4
CAtNIPP (trained on F_c=1)	460.5 ± 37.1	470.8 ± 38.3	485.6 ± 38.6	450.4 ± 40.9	465.2 ± 38.6	483.7 ± 41.1	442.0 ± 37.9	464.1 ± 42.7	479.3 ± 45.5
CAtNIPP (trained on F_c=5)	428.2 ± 28.5	427.0 ± 31.8	429.2 ± 34.4	367.2 ± 27.7	372.9 ± 32.2	380.2 ± 37.4	352.8 ± 26.7	367.9 ± 30.4	375.6 ± 36.4
CAtNIPP (trained on F_c=10)	496.4 ± 51.4	498.8 ± 52.2	498.6 ± 50.0	464.8 ± 51.9	471.3 ± 55.5	481.7 ± 53.8	457.8 ± 53.2	469.6 ± 54.9	473.5 ± 50.9
CAtNIPP + DR	419.3 ± 30.3	422.8 ± 29.3	427.5 ± 32.6	378.7 ± 25.9	387.1 ± 35.9	392.7 ± 37.0	376.4 ± 28.7	389.3 ± 35.3	393.1 ± 37.0
DyPNIPP (Ours)	**401.3 ± 25.5**	**403.5 ± 26.4**	**403.2 ± 27.3**	**349.1 ± 25.3**	**349.7 ± 24.9**	**348.9 ± 26.5**	**340.9 ± 23.0**	**342.8 ± 27.1**	**347.3 ± 28.7**

Table 1: Covariance Trace comparison across 200 instances for three budgets; lower values indicate better performance.

Model used	Environment Parameter (Fuel and Vegetation coefficient)								
	budget = 7			budget = 11			budget = 15		
	F_c=1	F_c=5	F_c=10	F_c=1	F_c=5	F_c=10	F_c=1	F_c=5	F_c=10
Sampling-based IPP (non-RL)	7.2 ± 6.8	9.5 ± 6.5	10.9 ± 7.9	12.5 ± 14.1	14.2 ± 10.9	16.0 ± 12.8	14.1 ± 14.8	16.8 ± 13.6	17.9 ± 14.2
CAtNIPP (trained on F_c=1)	6.2 ± 1.3	13.4 ± 3.1	9.5 ± 2.2	9.5 ± 2.0	13.2 ± 2.4	16.1 ± 4.1	13.7 ± 2.6	21.8 ± 4.8	25.0 ± 5.3
CAtNIPP (trained on F_c=5)	6.3 ± 1.3	8.2 ± 1.6	9.0 ± 2.1	9.4 ± 1.5	12.1 ± 2.1	13.6 ± 2.7	15.0 ± 2.7	18.7 ± 2.8	18.7 ± 2.8
CAtNIPP (trained on F_c=10)	6.2 ± 1.5	8.4 ± 2.1	9.7 ± 2.6	10.2 ± 2.7	13.3 ± 3.4	15.4 ± 4.1	16.2 ± 3.6	21.7 ± 4.7	20.9 ± 3.0
CAtNIPP + DR	6.1 ± 1.3	8.2 ± 1.6	9.9 ± 2.0	9.1 ± 1.7	12.9 ± 2.7	15.3 ± 3.5	14.9 ± 2.4	21.4 ± 4.6	24.0 ± 5.6
DyPNIPP (Ours)	**5.2 ± 1.0**	**6.9 ± 1.4**	**7.8 ± 1.5**	**7.7 ± 1.3**	**10.9 ± 1.9**	**11.5 ± 2.3**	**10.3 ± 1.6**	**14.0 ± 2.6**	**14.7 ± 2.9**

Table 2: Cumulative RMSE comparison across 200 instances for three budgets; lower values indicate better performance.

domly sampled between $[0, 10]$ for every episode (referred to as CAtNIPP+DR in Table 2). Since CAtNIPP was originally designed for static, time-invariant environments, we extend it to spatio-temporal settings by including time as a regressor for the GP. Lastly, we evaluate a sampling-based informative planner, where the next target location is chosen by maximizing predictive entropy minus a travel-distance penalty. To initialize the spatio-temporal GP for this method, we provide 100 randomly sampled initial observations.

We compare DyPNIPP with these baselines across three environments with $F_c \in \{1, 5, 10\}$ and three different budgets ($B \in \{7, 11, 15\}$). We use two metrics: covariance trace $Tr(P)$ and cumulative RMSE. The covariance trace measures the remaining uncertainty of the GP model, aligning with the IPP objective in Sec. 2.3. However, remaining uncertainty does not necessarily imply the accuracy of predicted environmental phenomena, so we additionally measure cumulative RMSE between predicted environment evolution and ground truth over the trajectory. Results are shown in Table 1 and Table 2. We observe the following:

• **The prior RL-based IPP algorithm lacks robustness**: CAtNIPP trained on a specific F_c exhibits suboptimal performance in environments with a different F_c. For example, in the case of a budget of 15, CAtNIPP trained on $F_c = 1$ outperforms CAtNIPP trained on $F_c = 5$ and $F_c = 10$ in an environment with $F_c = 1$, whereas it performs poorly in environments with $F_c = 5$ and $F_c = 10$.

• **Domain randomization alone is insufficient**: CAtNIPP+DR performs intermediate to the CAtNIPP policies trained on $F_c = 1, 5, 10$. This implies that DyPNIPP without environment dynamic prediction converges to the optimal policy for averaged environment dynamics, which is suboptimal for each individual environment dynamic.

• **DyPNIPP is robust under dynamic process variation**: DyPNIPP outperforms the baselines on both metrics. In addition, DyPNIPP shows similar covariance trace performance regardless of

the underlying fuel distribution. These are strong pieces of evidence supporting the **robustness** of our approach against variations in environment dynamics.

4.3.2. PERFORMANCE COMPARISON OF VARIATION IN NUMBER OF FIRES

We additionally study how the policy behaves when the number of simultaneously propagating fires varies. Each fire originates at a randomly sampled timestamp within an episode. We evaluate DyPNIPP and CAtNIPP+DR (both trained with up to 3 fires) on environments with 1, 3, and 5 fires, and report cumulative RMSE. Note that the 5-fire case is out-of-distribution for both policies. As shown in Table 4, DyPNIPP consistently outperforms CAtNIPP+DR across all settings, demonstrating robustness not only to variation in fuel coefficient but also to variation in the number of fires.

Model used	Environment Parameter (Fuel coefficient)								
	n(fires) = 1			n(fires) = 3			n(fires) = 5		
	F_c=1	F_c=5	F_c=10	F_c=1	F_c=5	F_c=10	F_c=1	F_c=5	F_c=10
CAtNIPP	10.3±	14.5±	16.5±	13.9±	16.5±	19.7±	14.0±	18.3±	20.5±
+ DR	2.3	3.0	3.4	2.6	3.6	2.9	2.6	3.1	3.7
DyPNIPP	**9.3±**	**11.9±**	**13.0±**	**12.2±**	**14.4±**	**15.0±**	**13.4±**	**15.4±**	**15.5±**
(Ours)	**2.0**	**3.1**	**3.3**	**2.6**	**3.6**	**3.8**	**2.7**	**3.4**	**3.2**

Figure 4: Cumulative RMSE comparison across environments with different numbers of fire origins. All policies are trained in environments with up to 3 fires.

4.3.3. PERFORMANCE COMPARISON IN AIR TEMPERATURE PREDICTION

We additionally evaluate DyPNIPP on a real-world air temperature dataset in which environment dynamics naturally evolve within an episode. This setting tests robustness under unstructured, real-world variations rather than parameterized shifts. As shown in Table 3, DyPNIPP achieves lower covariance trace than CAtNIPP, showing improved performance under naturally varying environmental phenomena.

Model used	Covariance Trace		
	budget=4	budget=7	budget=10
CAtNIPP	238.7±42.9	133.3±32.9	109.8±28.8
DyPNIPP (Ours)	**215.7 ± 44.1**	**126.35 ± 21.8**	**102.7 ± 15.4**

Table 3: Covariance Trace comparison in air temperature domain; lower values indicate better performance.

4.4. Analysis: Dynamics Prediction Model

We design the DPM to predict the belief of the next observation so that it can capture the underlying environment dynamics. To verify this design choice, we conduct an ablation study comparing two modified versions of DyPNIPP that instead predict: (1) the belief of the current observation, and (2) the belief difference between the current and next observations. As shown in Table 4, predicting the current observation yields significantly worse performance, since it does not encode information about how the environment evolves. Predicting the belief difference performs better, but remains slightly below the full DyPNIPP, confirming that forecasting the next observation is more effective for learning a dynamics-aware representation. We additionally provide visualization results of the DPM latent embeddings across environments with different fuel coefficients, showing that the learned representations vary with the underlying environment dynamics, provided in the Appendix B.

Prediction	Environment Parameter					
	budget=7			budget=11		
	F_c=1	F_c=5	F_c=10	F_c=1	F_c=5	F_c=10
$b(o_t)$	7.2 ± 1.2	8.9 ± 1.5	9.9 ± 1.5	10.8 ± 1.6	13.7 ± 1.9	14.4 ± 2.1
$b(o_{t+1})$-$b(o_t)$	5.3 ± 1.1	6.8 ± 1.4	7.9 ± 1.5	8.0 ± 1.4	10.7 ± 1.9	11.7 ± 2.3
$b(o_{t+1})$ **(Ours)**	5.2 ± 1.0	7.0 ± 1.4	7.8 ± 1.5	7.8 ± 1.3	10.9 ± 1.9	11.5 ± 2.3

Table 4: Ablation on dynamics prediction model design.

4.5. Analysis: Planning Time

A key advantage of using RL for IPP is reduced planning time, which is critical for real-time deployment. Using a graph size of 200 with a budget of 15 units, DyPNIPP, CAtNIPP, and the sampling-based non-RL IPP method require 2.60 ± 0.26, 1.92 ± 0.29, and 18.05 ± 12.90 seconds respectively to compute the full path. While DyPNIPP incurs a slightly higher cost than CAtNIPP due to the additional network components, it remains faster than the sampling-based method.

4.6. Experimental Validation on Real Robot

We provide experimental validation on a real robot to verify that our IPP policy, trained entirely in simulation, can be deployed in the real world under physical sensing and execution noise. We project the spatio-temporal wildfire environment onto a physical 1.5 m $\times$ 1.5 m arena, maintaining consistency in configuration between simulation and deployment. We use a Khepera-IV robot equipped with a Raspberry Pi 3 and camera module. The robot observes the fire intensity only at its current location, updates its belief via GP regression, and the updated belief is then used as input to the policy at each step. A budget of 12 m (equivalent to 8 units in the unit grid) is used. As shown in Fig. 5, the learned policy successfully finds an informative path in the real environment, with a decision-making time less than 0.25 s on an Intel Xeon CPU. This result empirically demonstrates robustness under unmodeled real-world dynamics and validates that DyPNIPP can be deployed for real-time planning in spatio-temporal environments. The full video can be accessed here.

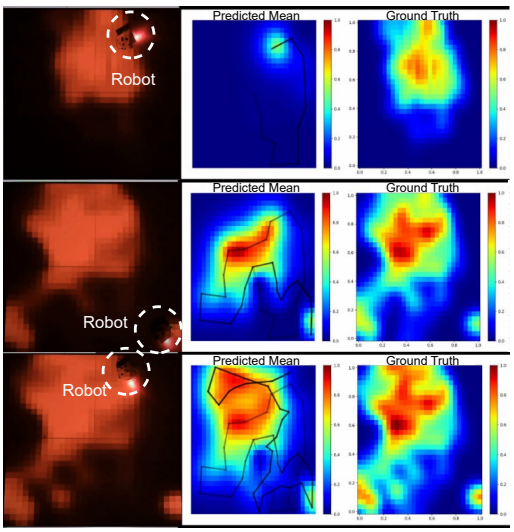

Figure 5: Experimental validation of DyPNIPP on the Khepera-IV robot. Left: the robot executing informative path planning in the physical arena (red indicates higher intensity). Right: predicted environment vs. ground truth over time (top to bottom), where prediction quality improves as more observations are collected.

5. Conclusion

In this paper, we proposed DyPNIPP, an RL-based framework for informative path planning that is robust to variations in spatio-temporal environment dynamics. DyPNIPP incorporates domain randomization to expose the agent to diverse dynamics during training, and introduces a dynamics prediction model that infers the current environment dynamics by predicting the belief of the next observation. We demonstrate that DyPNIPP achieves superior performance compared to existing IPP methods across diverse dynamic settings, and additionally validate real-time deployment on a physical robot, highlighting its potential for real-world applications. The Appendix is available at: here.

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.

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.

Yuhong Cao, Rui Zhao, Yizhuo Wang, Bairan Xiang, and Guillaume Sartoretti. Deep reinforcement learning-based large-scale robot exploration. *IEEE Robotics and Automation Letters*, 2024.

Jongseong Chae, Seungyul Han, Whiyoung Jung, Myungsik Cho, Sungho Choi, and Youngchul Sung. Robust imitation learning against variations in environment dynamics. In *International Conference on Machine Learning*, pages 2828–2852. PMLR, 2022.

Esther Derman, Daniel J Mankowitz, Timothy A Mann, and Shie Mannor. Soft-robust actor-critic policy-gradient. *arXiv preprint arXiv:1803.04848*, 2018.

M. A. Finney. Farsite: Fire area simulator-model development and evaluation. *Res. Pap. RMRS-RP-4, Revised 2004. Ogden, UT: US Department of Agriculture, Forest Service, Rocky Mountain Research Station. 47 p.*, vol. 4, 1998.

Srikar Babu Gadipudi, Srujan Deolasee, Siva Kailas, Wenhao Luo, Katia Sycara, and Woojun Kim. Offripp: Offline rl-based informative path planning. *arXiv preprint arXiv:2409.16830*, 2024.

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.

Kalvik Jakkala and Srinivas Akella. Multi-robot informative path planning from regression with sparse gaussian processes. *arXiv preprint arXiv:2309.07050*, 2023.

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.

Eugenia Kalnay, Masao Kanamitsu, Robert Kistler, William Collins, Dennis Deaven, Lev Gandin, Mark Iredell, Suranjana Saha, Glenn White, John Woollen, et al. The ncep/ncar 40-year reanalysis project. In *Renewable energy*, pages Vol1_146–Vol1_194. Routledge, 2018.

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

L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996. doi: 10.1109/70.508439.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 5757–5766. PMLR, 2020.

Jingsong Liang, Zhichen Wang, Yuhong Cao, Jimmy Chiun, Mengqi Zhang, and Guillaume Adrien Sartoretti. Context-aware deep reinforcement learning for autonomous robotic navigation in unknown area. In *Conference on Robot Learning*, pages 1425–1436. PMLR, 2023.

Daniel Mankowitz, Timothy Mann, Pierre-Luc Bacon, Doina Precup, and Shie Mannor. Learning robust options. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Daniel J Mankowitz, Nir Levine, Rae Jeong, Yuanyuan Shi, Jackie Kay, Abbas Abdolmaleki, Jost Tobias Springenberg, Timothy Mann, Todd Hester, and Martin Riedmiller. Robust reinforcement learning for continuous control with model misspecification. *arXiv preprint arXiv:1906.07516*, 2019.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

Anay Pattanaik, Zhenyi Tang, Shuijing Liu, Gautham Bommannan, and Girish Chowdhary. Robust deep reinforcement learning with adversarial attacks. *arXiv preprint arXiv:1712.03632*, 2017.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

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.

Marija Popovic, Joshua Ott, Julius Rückin, and Mykel J Kochendorfer. Robotic learning for adaptive informative path planning. *arXiv preprint arXiv:2404.06940*, 2024.

Julius Rückin, Liren Jin, and Marija Popović. Adaptive informative path planning using deep reinforcement learning for uav-based active sensing. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4473–4479. IEEE, 2022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

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

Esmaeil Seraj, Xiyang Wu, and Matthew Gombolay. Firecommander: An interactive, probabilistic multi-agent environment for heterogeneous robot teams. *arXiv preprint arXiv:2011.00165*, 2020.

Reda Bahi Slaoui, William R Clements, Jakob N Foerster, and Sébastien Toth. Robust visual domain randomization for reinforcement learning. *arXiv preprint arXiv:1910.10537*, 2019.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

Gabriele Tiboni, Pascal Klink, Jan Peters, Tatiana Tommasi, Carlo D'Eramo, and Georgia Chalvatzaki. Domain randomization via entropy maximization, 2023.

Apoorva Vashisth, Julius Rückin, Federico Magistri, Cyrill Stachniss, and Marija Popović. Deep reinforcement learning with dynamic graphs for adaptive informative path planning. *arXiv preprint arXiv:2402.04894*, 2024.

Yizhuo Wang, Yutong Wang, Yuhong Cao, and Guillaume Sartoretti. Spatio-temporal attention network for persistent monitoring of multiple mobile targets. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3903–3910. IEEE, 2023.

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*, pages 31–44. Springer, 2016.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in Neural Information Processing Systems*, 33:21024–21037, 2020.

Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pages 737–744. IEEE, 2020.