Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

POAM: Probabilistic Online Attentive Mapping for Efficient Robotic Information Gathering

Weizhe Chen, Lantao Liu and Roni Khardon
Luddy School of Informatics, Computing, and Engineering
Indiana University, Bloomington, IN 47408, USA
Emails: chenweiz@iu.edu, lantao@iu.edu, rkhardon@iu.edu

***Abstract*—Gaussian Process (GP) models are widely used for Robotic Information Gathering (RIG) in exploring unknown environments due to their ability to model complex phenomena with non-parametric flexibility and accurately quantify prediction uncertainty. Previous work has developed informative planners and adaptive GP models to enhance the *data efficiency* of RIG by improving the robot's sampling strategy to focus on informative regions in non-stationary environments. However, *computational efficiency* becomes a bottleneck when using GP models in large-scale environments with limited computational resources. We propose a framework – Probabilistic Online Attentive Mapping (POAM) – that leverages the modeling strengths of the non-stationary Attentive Kernel while achieving *constant-time* computational complexity for online decision-making. POAM guides the optimization process via variational Expectation Maximization, providing constant-time update rules for inducing inputs, variational parameters, and hyperparameters. Extensive experiments in active bathymetric mapping tasks demonstrate that POAM significantly improves computational efficiency, model accuracy, and uncertainty quantification capability compared to existing online sparse GP models.**

I. Introduction

Robotic Information Gathering (RIG) is a fundamental research domain in robotics, focused on how robots can efficiently collect informative data to construct an accurate model of an unknown target function while adhering to robot embodiment constraints. RIG has broad applications, including autonomous environmental monitoring, active 3D reconstruction and inspection, and robotic exploration of uncharted environments [1, 18, 20, 28, 30, 34, 57, 58, 65, 67, 71].

To achieve effective active information acquisition, certain key features are desired in a model. Firstly, it should efficiently quantify uncertainty in its predictions to guide informed data collection, necessitating a probabilistic approach. Secondly, to facilitate real-time decision-making with limited on-board computational and memory resources, an online model is essential. Finally, the model should exhibit attentiveness in two specific ways: by directing the robot to focus on areas with high modeling errors, and by prioritizing the modeling of the most characteristic aspects of the target function, given the limited computational resources. Considering these requirements, we conclude that a *probabilistic*, *online*, and *attentive* model is necessary.

In spatial modeling, Gaussian Processes (GPs) are widely adopted due to their non-parametric flexibility and precise uncertainty quantification capabilities [5, 14, 20, 21, 37, 55]. While previous research has primarily focused on developing informative planners and non-stationary kernels to enhance the *data efficiency* of RIG, there has been a significant gap in addressing *computation efficiency* [31, 32, 42]. This oversight presents a bottleneck in large-scale environments, especially when computational resources are limited.

In this work, we aim to address this critical gap by proposing an online and attentive GP model, building upon the recently introduced Attentive Kernel (AK) [13], which effectively guides robots to focus on informative regions. This paper tackles the computational limitations of standard GP regression, enabling long-term, long-range RIG missions. Our goal is to achieve *constant-time* computational complexity, which is essential for online decision making, without compromising prediction accuracy and precision in uncertainty quantification.

To address the computational challenge, sparse Gaussian Process (GP) methods are widely used to scale GP models for handling large-scale problems effectively [7, 26, 54]. While numerous efforts aim to reduce the computational cost of GPs, existing methods often fail when the data or kernel are non-stationary or when data is collected sequentially in a stream, as in RIG. Initially, we attempted to integrate the AK with established online SGP methods, such as Streaming Sparse GPs (SSGP) introduced by Bui et al. [6] and Online Variational Conditioning (OVC) by Maddox et al. [43]. However, these approaches present several limitations that motivated us to develop a novel framework. Specifically, when these methods are used with the AK they face difficulties in adapting the input-dependent lengthscale, essentially degenerating to a stationary kernel. SSGP also has suboptimal runtime performance due to the complexity of its online objective function calculation. Additionally, the numerical stability of this method was occasionally compromised, leading to Cholesky decomposition errors. With OVC, the critical aspect of hyperparameter optimization is absent from their discussion, posing a gap in understanding and potentially impacting the overall efficacy of the method.

To overcome these limitations, we introduce the Probabilistic, Online, and Attentive Mapping (POAM) framework for learning SGP models with the AK. Our analysis reveals that the primary cause of the existing methods' inability to properly learn the lengthscale is the interplay between the optimization of inducing inputs and the kernel lengthscale within SGP

methods. To address this, we propose a solution that strategically guides the optimization process by focusing solely on learning the lengthscale and directly computing the inducing-point locations using a lengthscale-dependent strategy. Furthermore, we recognize that optimizing inducing inputs, variational parameters, and hyperparameters should be orchestrated within a variational Expectation-Maximization (EM) framework. For efficient online decision-making, we introduce a constant-time update rule for variational parameters, leveraging the additive property of the data-dependent variables in the posterior distribution. Additionally, for hyperparameter optimization, our approach employs mini-batch stochastic optimization on the entire dataset, thereby bypassing the extra computational cost associated with computing the Kullback-Leibler (KL) divergence in an online objective function.

In summary, the contributions of this work are threefold:

- We diagnose the failure of SGP methods with non-stationary kernels like AK, providing insights for learning input-dependent lengthscales.
- We develop constant-time update rules for inducing inputs, variational parameters, and hyperparameters in the SGP framework, enabling real-time decision-making.
- We show that the proposed POAM framework outperforms existing methods in modeling accuracy, uncertainty quantification, and computational efficiency in active bathymetric mapping tasks. We also open-source the code for future research[1].

II. Related Work

Robotic Information Gathering Mobile robots have been widely utilized as autonomous data-collecting devices, significantly advancing scientific research, especially in remote hazardous environments [18, 35, 38]. These robots find applications in diverse domains, including environmental mapping and monitoring [22, 25, 29, 44, 64], search and rescue missions [46, 48, 51], and 3D reconstruction projects [36, 69, 71].

In RIG research, the primary focus is on informative planning – strategically planning action sequences or learning policies to acquire valuable data [53]. This emphasis has led to various planners, including recursive greedy approaches [4, 47, 61], dynamic programming methods [10, 40], mixed-integer quadratic programming solutions [68], sampling-based algorithms [1, 3, 28, 30, 34, 49, 57, 59], and trajectory optimization techniques [2, 16, 45]. Additionally, efforts have been made to enhance the computational efficiency of information-theoretic objective functions in RIG [11, 12, 67, 70]. Last but not least, there is a small but growing body of work focusing on the development of probabilistic models and their multi-robot extensions for RIG [31–33, 41, 42, 52]. An independent yet closely related research direction is *ergodic search*, which plans a continuous sampling trajectory such that the time a robot spends in a region is generally proportional to the information available in that region [17, 56, 63].

[1] https://github.com/Weizhe-Chen/POAM

Online Sparse Gaussian Processes Sparse Gaussian process reduce the computational complexity of GP by performing inference through a reduce set of data points or pseudo data points [24, 54, 60, 66]. For the online setting Csató and Opper [15] employed Expectation Propagation (EP) for inference and used a projection method to achieve sparsity, but they did not estimate hyperparameters. In contrast, Bui et al. [6] introduced Streaming Sparse Gaussian Processes (SSGP), which are capable of estimating hyperparameters through an online evidence lower bound (ELBO). They observed that EP performed worse than variational inference (VI). A different perspective is provided by Maddox et al. [43], who interpreted SSGP as Sparse Gaussian Process Regression (SGPR) on an augmented dataset. This interpretation simplifies derivations and implementation but does not address hyperparameter optimization. Additionally, Stanton et al. [62] enhanced the computational efficiency of SGPR by adopting a structured covariance function, which may limit the model's flexibility.

Relationship Between Existing Work and Our Work Our work is closely related to SSGP [6] and OVC [43]. Compared to SSGP, in addition to the difference in optimization objective, our method has three key differences. First, we use a different update rule for variational parameters, leading to improved performance and numerical stability. Second, while SSGP initializes inducing inputs by randomly sampling from old inducing inputs and new training inputs and then optimizes the inducing point locations using the online ELBO, we employ pivoted Cholesky decomposition for inducing point selection without further optimization. This approach is faster and avoids training difficulties. Third, for hyperparameter optimization, SSGP optimizes the online ELBO on new batches until convergence and discards the data. In contrast, we sample mini-batches from the entire dataset and use stochastic optimization for a fixed number of iterations, leveraging the fact that RIG is not a streaming problem and all data is available for training. Compared to OVC, our method has two differences: it saves different data-dependent terms for updating variational parameters, resulting in significantly better performance in the experiments, and it addresses hyperparameter optimization, a crucial aspect for RIG that OVC does not discuss extensively.

III. Background

A. Problem Formulation

We consider a regression problem where the robot observes training data $\mathbb{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$. Here, $\mathbf{x}_i \in \mathbb{R}^D$ is the i-th input and $y_i \in \mathbb{R}$ is the corresponding target value. We build a GP model using the training dataset to predict the outputs $y_\star$ given any test input $\mathbf{x}_\star$, while taking into account the uncertainty about the prediction for active information gathering. The number of training data N grows as the robot collects more data. Our goal is to develop an algorithm that can efficiently update the GP model and make predictions in *constant* time for online decision making.

B. Attentive Kernel (AK)

We use the Attentive Kernel (AK) [13], which is *non-stationary* and has been shown to provide improved prediction accuracy and uncertainty quantification compared to commonly used stationary kernels. The kernel is defined as:

$$k(\mathbf{x}, \mathbf{x}') = \alpha \bar{\mathbf{w}}^\intercal \bar{\mathbf{w}}' \sum_{m=1}^{M} \bar{w}_m \bar{w}'_m k_m(\mathbf{x}, \mathbf{x}'),$$

where α is the kernel amplitude, $\bar{\mathbf{w}} = {}^{\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})}/_{\|\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})\|_2}$ is the normalized weight vector, and $\bar{w}_m$ is its m-th element. The vector-valued function $\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \mapsto [0,1]^M$ is the output of a neural network parameterized by $\boldsymbol{\theta}$.

We use Gaussian kernels with fixed lengthscales ℓ_m equidistant spaced from ℓ_{min} to ℓ_{max} as the base kernels:

$$k_m(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2{\ell_m}^2}\right).$$

AK captures mild spatially-varying variability by adaptively blending multiple base kernels with different length-scales via input-dependent weights $\bar{w}_m \bar{w}'_m$. It also handles sharp transitions by zeroing out the correlation between two points through the dot product of their membership vectors $\bar{\mathbf{w}}^\intercal \bar{\mathbf{w}}'$. The parameters of the weighting function $\mathbf{w}_{\boldsymbol{\theta}}$ are learned from data by maximizing the standard marginal likelihood or the evidence lower bound.

Compared to commonly used stationary kernels in RIG, AK can better capture key characteristic patterns of the underlying environment and provide more informative uncertainty estimates to guide the robot's sampling process, making it particularly suitable for our problem.

C. Gaussian Process Regression (GPR)

Model In Gaussian process regression (GPR), the target values y are assumed to be the latent function values $f(\mathbf{x})$ corrupted by an additive Gaussian white noise:

$$y = f(\mathbf{x}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(\varepsilon \mid 0, \sigma^2).$$

A zero-mean GP *prior* is placed over the latent function:

$$f(\mathbf{x}) \sim \mathcal{GP}(0, k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')), \tag{1}$$

where $k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')$ is the covariance function, a.k.a. kernel function, with kernel parameters $\boldsymbol{\theta}$.

Prediction The GP prior and the Gaussian likelihood are *conjugate*, which yields an elegant closed-form solution for the predictive distribution of the latent function $f_\star$ at any test input $\mathbf{x}_\star$:

$$\begin{aligned} p(f_\star \mid \mathbf{y}) &= \mathcal{N}(f_\star \mid \mu, \nu), \\ \mu &= \mathbf{k}_{f\star}^\intercal \mathbf{K}_{yy}^{-1} \mathbf{y}, \\ \nu &= k_{\star\star} - \mathbf{k}_{f\star}^\intercal \mathbf{K}_{yy}^{-1} \mathbf{k}_{f\star}, \end{aligned} \tag{2}$$

where $\mathbf{k}_{f\star} \in \mathbb{R}^N$ is the kernel values between training inputs and the test input, $\mathbf{K}_{yy} = \mathbf{K}_{ff} + \boldsymbol{\Sigma}$ is the summation of the training covariance matrix $\mathbf{K}_{ff}$ and the diagonal observational noise matrix $\boldsymbol{\Sigma} = \sigma^2 \mathbf{I}$, and $k_{\star\star} \triangleq k(\mathbf{x}_\star, \mathbf{x}_\star)$.

Optimization The prediction quality of GPR depends on the settings of the *hyperparameters* $\boldsymbol{\phi} \triangleq [\alpha, \sigma, \boldsymbol{\theta}]$, which can be optimized via *model selection* by maximizing the *model evidence*, a.k.a. log *marginal likelihood*:

$$\begin{aligned} &\max_{\boldsymbol{\phi}} \ \log p_{\boldsymbol{\phi}}(\mathbf{y}) = \max_{\boldsymbol{\phi}} \ \log \mathcal{N}(\mathbf{y} \mid \mathbf{0}, \mathbf{K}_{yy}) \\ =& \max_{\boldsymbol{\phi}} \ -\frac{1}{2}\mathbf{y}^\intercal \mathbf{K}_{yy}^{-1} \mathbf{y} - \frac{1}{2}\log|\mathbf{K}_{yy}| - \frac{N}{2}\log(2\pi), \end{aligned} \tag{3}$$

where $|\mathbf{K}_{yy}|$ is the matrix determinant of $\mathbf{K}_{yy}$. We refer to hyperparameter optimization as *training* hereafter.

Complexity Training complexity of GPR is $\mathcal{O}(N^3)$ due to the inversion and determinant computation of $\mathbf{K}_{yy}$ and prediction complexity is $\mathcal{O}(N^2)$ per test input due to matrix multiplications.

D. Sparse Gaussian Process Regression (SGPR)

Model Sparse Gaussian process regression (SGPR) alleviates the computational burden by approximating the full GP model with a sparse model that shifts the expensive computations to a small set of *inducing points* [24, 54, 60, 66]. Specifically, the model is augmented by a small number of M *inducing points* $\{(\mathbf{z}_m, u_m)\}_{m=1}^M$, where *inducing outputs* $\mathbf{u} = [f(\mathbf{z}_1), \ldots, f(\mathbf{z}_M)]^\intercal$ are evaluated at corresponding *inducing inputs* $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_M]^\intercal$. The augmented model $p(\mathbf{y}, \mathbf{f}, \mathbf{u})$ can be factorized as

$$p(\mathbf{y}, \mathbf{f}, \mathbf{u}) = p(\mathbf{y} \mid \mathbf{f}) \ p(\mathbf{f} \mid \mathbf{u}) \ p(\mathbf{u}).$$

Assuming that the inducing outputs $\mathbf{u}$ effectively capture the information from the training outputs $\mathbf{f}$ (i.e., any other function value and $\mathbf{f}$ are *independent* given $\mathbf{u}$), the posterior predictive distribution of $f_\star$ can be expressed as:

$$\begin{aligned} p(f_\star \mid \mathbf{y}) &= \int p(f_\star \mid \mathbf{u}, \mathbf{f}) \ p(\mathbf{f} \mid \mathbf{u}, \mathbf{y}) \ p(\mathbf{u} \mid \mathbf{y}) \ \mathrm{d}\mathbf{f} \ \mathrm{d}\mathbf{u} \\ &= \int p(f_\star \mid \mathbf{u}) \ p(\mathbf{u} \mid \mathbf{y}) \ \mathrm{d}\mathbf{u}, \end{aligned}$$

which indicates that prediction can be made by only considering the inducing points.

In practice, however, the assumption that $\mathbf{u}$ serves as sufficient statistics is unlikely to hold, and the posterior distribution becomes intractable in non-conjugate cases. To address these issues, the posterior distribution is typically approximated by a structured variational distribution:

$$p(\mathbf{f}, \mathbf{u} \mid \mathbf{y}) \approx q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u}) \ q(\mathbf{u}),$$

where $p(\mathbf{f} \mid \mathbf{u})$ is the conditional prior and $q(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \mathbf{m}, \mathbf{S})$ is the approximate posterior of the inducing variables, assumed to follow a Gaussian distribution. The parameters $\mathbf{m}$ and $\mathbf{S}$ can be optimized via variational inference [24, 60, 66].

Prediction Choosing a Gaussian distribution for $q(\mathbf{u})$ leads to a closed-form solution for the predictive distribution $p(f_\star \mid \mathbf{y}) \approx q(f_\star)$:

$$\begin{aligned} q(f_\star) &= \mathcal{N}\left(f_\star \mid \tilde{\mu}, \tilde{\nu}\right), \\ \tilde{\mu} &= \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{m}, \\ \tilde{\nu} &= k_{\star\star} - \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{k}_{u\star} + \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{S} \mathbf{K}_{uu}^{-1} \mathbf{k}_{u\star}. \end{aligned} \tag{4}$$

Optimization Variational inference optimizes the parameters $\mathbf{Z}$, $\mathbf{m}$, and $\mathbf{S}$ by minimizing the Kullback-Leibler (KL) divergence, $\mathtt{KL}[q(\mathbf{f},\mathbf{u})\|p(\mathbf{f},\mathbf{u}\,|\,\mathbf{y})]$. This is equivalent to maximizing the evidence lower bound (ELBO) [66]:

$$\log p(\mathbf{y}) \geq \int q(\mathbf{f},\mathbf{u})\ \log \frac{p(\mathbf{y},\mathbf{f},\mathbf{u})}{q(\mathbf{f},\mathbf{u})}\ \mathrm{d}\mathbf{f}\ \mathrm{d}\mathbf{u} \triangleq \mathtt{ELBO}, \tag{5}$$

$$\mathtt{ELBO} = \log \mathcal{N}(\mathbf{y}\mid \mathbf{0}, \mathbf{Q}_{yy}) - \frac{1}{2\sigma^2}\mathrm{tr}(\mathbf{K}_{ff}-\mathbf{Q}_{ff}). \tag{6}$$

Here, $\mathbf{Q}_{ff} = \mathbf{K}_{uf}^\intercal \mathbf{K}_{uu}^{-1}\mathbf{K}_{uf}$ and $\mathbf{Q}_{yy} = \mathbf{Q}_{ff} + \mathbf{\Sigma}$. This bound is referred to as the *collapsed* form of the ELBO, as the variational parameters $\mathbf{m}$ and $\mathbf{S}$ are analytically marginalized out due to the Gaussian likelihood used in the regression case.

The optimal variational parameters correspond to the collapsed ELBO are given by

$$\mathbf{m} = \frac{1}{\sigma^2}\mathbf{K}_{uu}\mathbf{A}^{-1}\mathbf{K}_{uf}\mathbf{y}, \tag{7}$$

$$\mathbf{S} = \mathbf{K}_{uu}\mathbf{A}^{-1}\mathbf{K}_{uu}, \text{ where} \tag{8}$$

$$\mathbf{A} = \mathbf{K}_{uu} + \frac{1}{\sigma^2}\mathbf{K}_{uf}\mathbf{K}_{uf}^\intercal. \tag{9}$$

Note that, although $\mathbf{m}$ and $\mathbf{S}$ are analytically marginalized out and thus do not require optimization, we still need to optimize the inducing inputs $\mathbf{Z}$ and the hyperparameters $\boldsymbol{\phi}$. This can be achieved simultaneously by maximizing the ELBO.

Complexity The matrix inversion operations in SGPR are $\mathcal{O}(M^3)$ instead of $\mathcal{O}(N^3)$ in GPR, which is no longer the bottleneck. The time complexity of SGPR is $\mathcal{O}(NM^2)$ due to the matrix multiplications, which is much lower than that of GPR when $M \ll N$.

E. Stochastic Variational Gaussian Processes (SVGP)

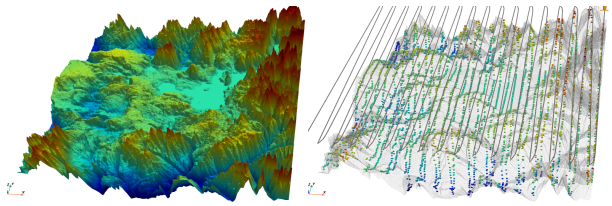
Optimization The linear complexity of SGPR can be further reduced to constant time using Stochastic Variational Gaussian Processes (SVGP) [24]. The ELBO in Eq. (5) before marginalizing out the variational parameters can be written in an *uncollapsed* form:

$$\mathtt{ELBO} = \sum_{n=1}^{N} \mathrm{E}_{q(f_n)}\left[\log p(y_n \mid f_n)\right] - \mathtt{KL}[q(\mathbf{u})\ \|\ p(\mathbf{u})], \tag{10}$$

where the marginal predictive distribution $q(f_n)$ is given by Eq. (4). The expectation can be computed analytically when the likelihood is Gaussian or approximated by Monte Carlo sampling otherwise [27]. Writing the ELBO as a sum of N terms allows us to use mini-batch stochastic optimization to update the parameters. Specifically, the gradient of the expected log-likelihood term can be approximated by B training data sampled uniformly at random:

$$\frac{N}{B}\sum_{b=1}^{B} \nabla \mathrm{E}_{q(f_b)}[\log p(y_b \mid f_b)]. \tag{11}$$

All the parameters, including the variational parameters $\mathbf{m}$ and $\mathbf{S}$, the inducing inputs $\mathbf{Z}$, and the hyperparameters $\boldsymbol{\phi}$, can be optimized by maximizing the uncollapsed ELBO in Eq. (10) using stochastic gradient ascent.



(a) Environment (b) Dataset

Fig. 1. **An elevation dataset sampled via lawnmower path in the environment for illustrative purposes.** (a) The ground-truth elevation map of the environment. Red and blue colors represent high and low elevations, respectively. (b) A dense dataset sampled by lawnmower path in the environment, which is used to train the GP model for elevation mapping.

Prediction The resulting variational parameters $\mathbf{m}$ and $\mathbf{S}$ can be directly plugged into Eq. (4) to make predictions, bypassing the need to compute them analytically from the training data, which would take linear time.

Complexity The time complexity becomes $\mathcal{O}(BM^2 + M^3)$ per training iteration and $\mathcal{O}(M^3)$ when making predictions on a test input, which is independent of the number of training data N.

IV. Methodology

Before introducing the complete Probabilistic, *Online*, and Attentive Mapping (POAM) framework, we first explain how to achieve probabilistic attentive mapping using SVGP with AK on the *entire* training dataset. Understanding this offline learning case is crucial for developing the online model. For illustrative purposes, we use the environment and dataset shown in Fig. 1 as a running example. The goal in this example is to make predictions that closely match the ground-truth environment using the collected data and to learn the underlying lengthscale map. Specifically, the left and center parts should have a large lengthscale, while the other three sides should have a small lengthscale. We then introduce the online update rules for the inducing inputs, variational parameters, and hyperparameters, enabling the model to be updated incrementally and efficiently as new data arrives.

A. Probabilistic Attentive Mapping with SVGP

Direct Use of SVGP with AK A straightforward solution to achieve probabilistic, online, and attentive mapping is to use an AK in SVGP and optimize all parameters using mini-batch stochastic optimization on an *ever-growing* dataset. Thus, directly training an SVGP with AK on the entire dataset is our initial attempt to achieving probabilistic attentive mapping. However, as shown in Fig. 2, this method encounters difficulties in learning the appropriate lengthscale map. The primary issue lies in the coupling of inducing input and kernel lengthscale optimization.

During the early stages of training, the hyperparameters of the AK have not yet been optimized and the input-dependent lengthscale behaves like a stationary kernel. At this stage, optimizing the ELBO in Eq. (10) results in inducing inputs

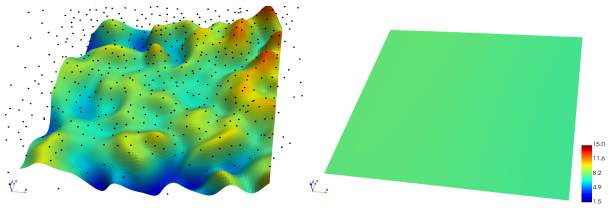
becoming uniformly distributed across the input space, encompassing the training data. When the optimization attempts to allocate a small lengthscale to a region with high variability, it requires a rapid relocation of more inducing inputs to that region. Failure to achieve this relocation incurs a high training loss because regions with small lengthscales require dense inducing-point support. However, gradient-based optimization restricts the movement of inducing inputs in each iteration, making it difficult to meet this requirement promptly. As a result, the optimizer tends to maintain a large lengthscale in AK even in regions of high variability. This ultimately leads to an inability to learn the input-dependent lengthscale, causing the model to degenerate into a stationary kernel, with inducing point locations becoming uninformative.

Pivoted Cholesky Decomposition Building upon the aforementioned observations, we introduce a novel strategy to guide the optimization process: *focusing solely on learning the lengthscale while directly computing the inducing input locations*. For the latter, we leverage Pivoted Cholesky Decomposition (PCD) [23], as proposed by Burt et al. [9] and used in OVC [43], to efficiently select inducing points from the training data.

Specifically, PCD computes a low-rank Cholesky factorization of a positive-definite matrix by iteratively selecting pivots to compute the permutation matrix for rearranging rows and columns. The algorithm starts with a residual matrix and a permutation matrix, which are initialized as the input matrix and an identity matrix, respectively. Iteratively, the algorithm selects pivots (the maximum diagonal elements of the residual matrix) to construct the Cholesky factor and update the residual matrix. This process continues until a specified rank is achieved or the remaining diagonal elements fall below a predefined error tolerance. Dynamic reordering of rows and columns using a permutation matrix derived from the selected pivot occurs throughout the iterations. With a computational complexity of $\mathcal{O}(NR^2)$, where R is the rank of the decomposition, PCD strikes a balance between numerical accuracy and efficiency.

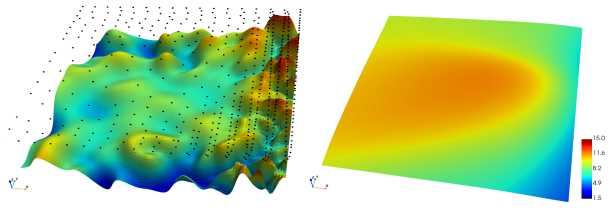
When applied to inducing input selection, the input is the kernel matrix $\mathbf{K}_{ff}$, and the output is the low-rank factor of $\mathbf{K}_{ff} \approx \mathbf{L}\mathbf{L}^\intercal$ along with the selected pivots. These pivots are then used to index the training data, enabling the selection of corresponding training inputs as the inducing inputs.

Intuitively, PCD allocates more inducing inputs in complex regions (small-lengthscale regions) and fewer inducing inputs in simple regions (large-lengthscale regions). This results in a more attentive inducing input distribution, enhancing the model's adaptability to spatially-varying complexities. Theory-wise, Burt et al. [9] derived upper bounds on the KL-divergence between the approximate posterior and the true posterior, which depend on either the *trace* or the *largest eigenvalues* of the low-rank Nyström approximation error. PCD provides a simple way to compute a low-rank approximation to a positive-definite matrix such that the trace error is rigorously controlled. It also enjoys exponential convergence rates when the eigenvalues of the full matrix exhibit a fast exponential decay [23]. These theoretical properties, along with the computational efficiency and numerical stability, make PCD a good choice for inducing input updates.



(a) SVGP Mean (b) SVGP Lengthscale

Fig. 2. **Prediction and lengthscale maps from jointly training all parameters with an Adam optimizer**. (a) The predictive mean map exhibits uniform smoothness due to the uniform scattering of inducing points (black dots) across the space and the failure to learn an input-dependent lengthscale. (b) The lengthscale map is flat, indicating that the attentive kernel has degenerated to a stationary kernel.



(a) PAM Mean (b) PAM Lengthscale

Fig. 3. **Prediction and lengthscale maps from the proposed Probabilistic Attentive Mapping (PAM) training paradigm**. (a) A higher density of inducing points is allocated to the complex region, enabling the predictive mean to capture finer elevation details. (b) The learned lengthscale map delineates the relatively smooth area on the left (large lengthscale) from the highly varying region near the right boundary (small lengthscale).

Analytic Computation of the Variational Parameters As shown in the ablation study in Section V-C, using PCD for inducing input selection and jointly optimizing all other parameters still results in poor performance. This is because the variational parameters $\mathbf{m}$ and $\mathbf{S}$ depend on the inducing inputs. When inducing inputs are updated instantaneously rather than being slowly optimized by gradient descent, the gradient optimization of the variational parameters cannot adjust accordingly, leading to poor performance. To resolve this, we also compute the variational parameters analytically using the closed-form expressions in Eqs. (7) and (8).

Variational Expectation Maximization Training To account for the interdependence of inducing inputs, variational parameters, and hyperparameters, we adopt a variational Expectation-Maximization (EM) approach to update these three sets of parameters. In the E-step, we fix the hyperparameters and sequentially update the inducing inputs and variational parameters. In the M-step, we fix the inducing inputs and variational parameters while optimizing the hyperparameters. This systematic updating process, when applied to SVGP with AK, is referred to as *Probabilistic Attentive Mapping (PAM)*.

The effectiveness of PAM is showcased in Fig. 3, where it demonstrates the ability to learn input-dependent lengthscales and strategically allocate more inducing inputs in high-

variability regions, leading to more accurate predictions.

However, while PAM excels in learning input-dependent lengthscales, its suitability for online learning is limited due to the non-incremental nature of variational parameter and inducing input computations. In the subsequent sections, we develop constant-time update rules specifically designed for inducing inputs and variational parameters.

B. Online Update of Inducing Inputs

Suppose we have M existing inducing inputs $\mathbf{Z}'$ selected from old training data $\mathbf{X}'$ and now receive N_{new} new training data $\mathbf{X} \in \mathbb{R}^{N_{\text{new}} \times D}$. Updating the inducing inputs by concatenating the old and new training data $[\mathbf{X'}^\intercal, \mathbf{X}^\intercal]^\intercal$ and running PCD again on the resulting $N \times N$ kernel matrix requires linear time complexity with respect to the number of training data N, which is not ideal for online learning.

To address this limitation, we opt for a more efficient *recursive* update rule for inducing inputs. Specifically, we concatenate the old inducing inputs and new training data $[\mathbf{Z'}^\intercal, \mathbf{X}^\intercal]^\intercal$ and perform PCD on the resulting square kernel matrix of size $M + N_{\text{new}}$, which is independent of the total number of training data. Note that $M + N_{\text{new}} \ll N$, and this recursive update rule allows us to efficiently update the inducing inputs in constant time complexity with respect to N, making it ideal for online learning.

C. Online Update of Variational Parameters

Recomputing the variational parameters using Eqs. (7) and (8) after receiving new data involves linear time complexity with respect to the number of training data. For real-time decision-making, we also need an incremental update strategy for the variational parameters. The key idea is to preserve the old data-dependent terms and compute only the new data-dependent terms related to the recently acquired data.

Denoting old variables with a prime symbol and explicitly writing out the data-dependent terms, we have:

$$\begin{bmatrix}\mathbf{K}_{uf'} & \mathbf{K}_{uf}\end{bmatrix}\begin{bmatrix}\mathbf{y}' \\ \mathbf{y}\end{bmatrix} = \underbrace{\mathbf{K}_{uf'}\mathbf{y}'}_{\text{old}} + \underbrace{\mathbf{K}_{uf}\mathbf{y}}_{\text{new}}, \tag{12}$$

$$\begin{bmatrix}\mathbf{K}_{uf'} & \mathbf{K}_{uf}\end{bmatrix}\begin{bmatrix}\mathbf{K}_{f'u} \\ \mathbf{K}_{fu}\end{bmatrix} = \underbrace{\mathbf{K}_{uf'}\mathbf{K}_{f'u}}_{\text{old}} + \underbrace{\mathbf{K}_{uf}\mathbf{K}_{fu}}_{\text{new}}. \tag{13}$$

The *additive* property of the data-dependent terms allows us to update the variational parameters incrementally. However, if we save the old data-dependent terms, $\mathbf{K'}_{u'f'}\mathbf{y}'$ and $\mathbf{K'}_{u'f'}\mathbf{K'}_{f'u'}$, they are computed using the old inducing inputs $\mathbf{u}'$ and hyperparameters $\boldsymbol{\phi}'$. To address this issue, we use a projection matrix to transform the saved terms to align with the new inducing inputs and hyperparameters. Formally, we would like to find $\mathbf{P} \in \mathbb{R}^{M \times M}$ such that

$$\mathbf{P}^\top \mathbf{K'}_{u'f'}\mathbf{y}' = \mathbf{K}_{uf'}\mathbf{y}', \tag{14}$$

$$\mathbf{P}^\top \mathbf{K'}_{u'f'}\mathbf{K'}_{f'u'}\mathbf{P} = \mathbf{K}_{uf'}\mathbf{K}_{f'u'}. \tag{15}$$

This requires solving the following linear system for $\mathbf{P}$:

$$\mathbf{P}^\top \mathbf{K'}_{u'f'} = \mathbf{K}_{uf'}, \text{ or equivalently, } \mathbf{K'}_{f'u'}\mathbf{P} = \mathbf{K}_{f'u}. \tag{16}$$

Since $\mathbf{K'}_{f'u'}$ is not a square matrix, the solution can be computed using the pseudo inverse:

$$\mathbf{P} = (\mathbf{K'}_{u'f'}\mathbf{K'}_{f'u'})^{-1}\mathbf{K'}_{u'f'}\mathbf{K}_{f'u}. \tag{17}$$

The computation of $\mathbf{K'}_{f'u'}$ and $\mathbf{K}_{f'u}$ requires all the old inputs $\mathbf{X}'$ before the current time step, which still requires linear time complexity with respect to the number of training data. To bypass this issue, we can choose a small set of "representative" old inputs for computing the kernel matrices. One computationally convenient choice is to use the old inducing inputs. By replacing $\mathbf{K'}_{f'u'}$ with $\mathbf{K'}_{u'u'}$ and $\mathbf{K}_{f'u}$ with $\mathbf{K}_{u'u}$ in Eq. (17), the projection matrix $\mathbf{P}$ is given by:

$$\mathbf{P} = (\mathbf{K'}_{u'u'}\mathbf{K'}_{u'u'})^{-1}\mathbf{K'}_{u'u'}\mathbf{K}_{u'u}, \tag{18}$$

$$= \mathbf{K'}^{-1}_{u'u'}\mathbf{K}_{u'u}. \tag{19}$$

Here, the first term is the inverse of the self-covariance matrix of the old inducing inputs, and the second term is the cross-covariance matrix between the old inducing inputs and the new inducing inputs.

In summary, we can update the variational parameters incrementally by adding the projected old data-dependent terms to the new data-dependent terms:

$$\mathbf{P}^\top \underbrace{\mathbf{K'}_{u'f'}\mathbf{y}'}_{\text{saved}} + \mathbf{K}_{uf}\mathbf{y}, \tag{20}$$

$$\mathbf{P}^\top \underbrace{\mathbf{K'}_{u'f'}\mathbf{K'}_{f'u'}}_{\text{saved}}\mathbf{P} + \mathbf{K}_{uf}\mathbf{K}_{fu}. \tag{21}$$

We then plug these updated terms into Eqs. (7) and (8) to update the variational parameters.

It is important to note that although Eqs. (20) and (21) resemble the projection-view of OVC, they actually store distinct sets of data-dependent terms: $\mathbf{K'}_{u'f'}\boldsymbol{\Sigma}^{-1}\mathbf{y}'$ and $\mathbf{K'}_{u'f'}\boldsymbol{\Sigma}^{-1}\mathbf{K'}_{f'u'}$. This difference results in significantly better performance of POAM in our experiments.

D. Online Update of Hyperparameters

For the online update of hyperparameters, our key insight is that RIG does not conform to a strict streaming problem. Typically, the robot retains access to all past data accumulated and stored onboard. To take advantage of this, we leverage the entire training set for hyperparameter updates while maintaining constant training complexity with respect to the number of training data.

Given that SVGP supports mini-batch stochastic optimization, we incorporate new data by appending it to the continually expanding training set. Hyperparameter optimization is then performed for a specified number of steps using Eq. (10), with gradients approximated through a Monte-Carlo estimate, as described in Eq. (11), computed from a mini-batch of randomly selected samples.

While this approach reduces the number of updates on the newly collected data because it may not be sampled in the mini-batch, this turns out to be advantageous in the context of RIG problems. During the initial phases, when the robot lacks sufficiently representative data of an unknown environment,

Algorithm 1: RIG with POAM

Collect $\mathbf{X}_0$ and $\mathbf{y}_0$ by following a pilot survey path;
Initialize $\mathbf{Z}_0$, $\mathbf{m}_0$, $\mathbf{S}_0$, and ϕ_0;
Decision epoch $t = 0$;
while *not collected enough number of samples* **do**
  Decision epoch $t = t + 1$;
  Select an informative waypoint;
  Collect samples $\mathbf{X}_t$ and $\mathbf{y}_t$ by navigating to the selected informative waypoints.;
  Concatenate $\mathbf{Z}_{t-1}$ and $\mathbf{X}_t$;
  Update $\mathbf{Z}_t$ via PCD on the kernel matrix computed on the concatenated inputs;
  Update the data-dependent terms using Eqs. (20) and (21);
  Update $\mathbf{m}_t$ and $\mathbf{S}_t$ via Eqs. (7) and (8);
  Optimize hyperparmeters ϕ on all data by mini-batch stochastic gradient ascent on Eq. (10);

frequent hyperparameter updates can prematurely lead the robot into an exploitative mode. Deliberately slowing down hyperparameter updates, especially for the lengthscale, allows the robot to initially explore the environment and gather more representative data for refining its model.

E. Probabilistic, Online, and Attentive Mapping (POAM)

The overarching POAM framework is outlined in Algorithm 1. Initially, the robot conducts a pilot survey to collect training data, which is essential for computing normalizing statistics for data preprocessing and initializing the GP model. To achieve this, the initial path should cover various locations within the workspace boundaries to gather representative samples. During each decision epoch, the robot selects informative waypoints and collects new samples while navigating to these waypoints. The newly acquired training inputs, along with the previous inducing inputs, are then used to update the inducing inputs via PCD (Section IV-B). Once the inducing inputs are updated, the variational parameters are adjusted using the online updates described in Section IV-C. Finally, hyperparameters are optimized by performing mini-batch stochastic gradient ascent on the ELBO for several steps (Section IV-D). Overall, POAM enables accurate and efficient probabilistic mapping of the unknown target function.

V. Experiments

We evaluate the proposed POAM framework's accuracy, uncertainty quantification capability, and computational efficiency through extensive experiments. In benchmarking, POAM is compared with two state-of-the-art online sparse GP models or their enhanced versions. The results, both quantitative and qualitative, are discussed in Section V-B. Additionally, an ablation study is conducted to validate the importance of the POAM components, detailed in Section V-C.

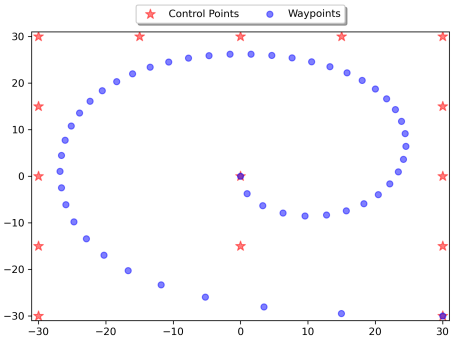


Fig. 4. **Illustration of the initial waypoints generated by a Bézier curve**.

A. Experiment Setup

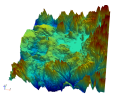
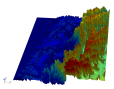
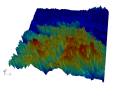
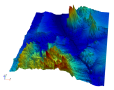
Task Setting Consider an Autonomous Underwater Vehicle (AUV) following a Dubins' car kinematic model on a fixed altitude plane, with a maximum linear velocity of 1 m/s and a control frequency of 10 Hz. The AUV is equipped with a single-beam range sensor that collects noisy elevation measurements at 3 Hz, with unit Gaussian white noise. The probabilistic model's inputs are two-dimensional sampling locations, and it can predict elevation at any query location along with the corresponding prediction uncertainty. In this active bathymetric mapping task, the AUV aims to minimize elevation prediction error efficiently, given a budget of 5000 samples. A superior method should achieve a lower prediction error by the end of the task and show a faster reduction in the prediction error curve. For simplicity, all compared methods use the same planner proposed in [13]. This planner evaluates prediction entropy at 2000 random candidate locations and selects a point maximizing high-entropy and minimizing distance to the current location of the robot as the next informative waypoint. As discussed in Section V-D the conclusions of the experiments are still valid when the planner is changed.

Initial Sampling The robot initiates its trajectory by following a Bézier curve produced by 15 control points that adapt to the dimensions of the workspace, as illustrated in Fig. 4. It is important to note that the pilot survey path is not limited to this Bézier curve, and the robot starting from the lower-right is an arbitrary choice; any initial path that helps the robot collect representative initial samples can be used. Experiments showing robustness to different initialization strategies are discussed in Section V-D. These samples initialize the GP model and compute the statistics needed to normalize the input values to fall within the range of -1 to 1 and standardize the target values to have a mean of 0 and a standard deviation of 1. This preprocessing step enhances training efficacy and numerical stability, which is a common practice in GPR.

Environment Description We use digital elevation maps from the NASA Shuttle Radar Topography Mission (SRTM) [19] to simulate the ground-truth environments for

TABLE I

BENCHMARKING RESULTS. THE TABLE SHOWS THE AVERAGE PERFORMANCE ACROSS ALL DECISION EPOCHS FOR THE PROPOSED POAM AND THE COMPARED BASELINES IN FOUR DIFFERENT ENVIRONMENTS. METRICS INCLUDE SMSE, MSLL, AND TRAINING TIME. THE BEST-PERFORMING METHOD IS HIGHLIGHTED IN BOLD. RESULTS ARE AVERAGED OVER 10 RUNS, WITH THE STANDARD DEVIATION INDICATED BY THE PLUS-MINUS SIGN. SUPERSCRIPTS AND SUBSCRIPTS DENOTE MAXIMUM AND MINIMUM VALUES, WHILE ARROWS INDICATE THE DIRECTION OF IMPROVEMENT.

Name	Environment	Method	Averaged SMSE$\downarrow_0^1$	Averaged MSLL$\downarrow^0$	Averaged Time (s)$\downarrow_0$
Env1		SSGP++	$(3.22 \pm 0.17) \times 10^{-1}$	$(-6.52 \pm 0.78) \times 10^{-1}$	$(1.57 \pm 0.02) \times 10^{0}$
		OVC	$(3.18 \pm 0.11) \times 10^{-1}$	$(-6.16 \pm 0.18) \times 10^{-1}$	1.18×10^{0}
		OVC++	$(3.13 \pm 0.21) \times 10^{-1}$	$(-6.70 \pm 0.24) \times 10^{-1}$	$\mathbf{(8.51 \pm 0.10) \times 10^{-1}}$
		POAM	$\mathbf{(3.09 \pm 0.11) \times 10^{-1}}$	$\mathbf{(-7.08 \pm 0.11) \times 10^{-1}}$	$\mathbf{(8.56 \pm 0.09) \times 10^{-1}}$
Env2		SSGP++	$(8.40 \pm 0.45) \times 10^{-2}$	$(-1.34 \pm 0.04) \times 10^{0}$	$(1.57 \pm 0.02) \times 10^{0}$
		OVC	$(9.35 \pm 0.38) \times 10^{-2}$	$(-1.13 \pm 0.01) \times 10^{0}$	$(1.18 \pm 0.01) \times 10^{0}$
		OVC++	$(8.39 \pm 0.20) \times 10^{-2}$	$(-1.30 \pm 0.01) \times 10^{0}$	$\mathbf{(8.53 \pm 0.12) \times 10^{-1}}$
		POAM	$\mathbf{(7.73 \pm 0.28) \times 10^{-2}}$	$\mathbf{(-1.46 \pm 0.04) \times 10^{0}}$	$\mathbf{(8.43 \pm 0.06) \times 10^{-1}}$
Env3		SSGP++	$(1.68 \pm 0.11) \times 10^{-1}$	$(-1.06 \pm 0.05) \times 10^{0}$	$(1.58 \pm 0.03) \times 10^{0}$
		OVC	$(1.74 \pm 0.06) \times 10^{-1}$	$(-8.82 \pm 0.13) \times 10^{-1}$	$(1.18 \pm 0.01) \times 10^{0}$
		OVC++	$(1.72 \pm 0.05) \times 10^{-1}$	$(-1.03 \pm 0.04) \times 10^{0}$	$\mathbf{(8.50 \pm 0.12) \times 10^{-1}}$
		POAM	$\mathbf{(1.62 \pm 0.09) \times 10^{-1}}$	$\mathbf{(-1.14 \pm 0.02) \times 10^{0}}$	$\mathbf{(8.46 \pm 0.11) \times 10^{-1}}$
Env4		SSGP++	$(9.79 \pm 2.34) \times 10^{-2}$	$(-1.22 \pm 0.16) \times 10^{0}$	$(1.58 \pm 0.02) \times 10^{0}$
		OVC	$\mathbf{(7.09 \pm 0.29) \times 10^{-2}}$	$(-1.28 \pm 0.02) \times 10^{0}$	$(1.18 \pm 0.01) \times 10^{0}$
		OVC++	$(1.06 \pm 0.26) \times 10^{-1}$	$(-1.15 \pm 0.14) \times 10^{0}$	$\mathbf{(8.59 \pm 0.10) \times 10^{-1}}$
		POAM	$(8.30 \pm 1.16) \times 10^{-2}$	$\mathbf{(-1.42 \pm 0.05) \times 10^{0}}$	$\mathbf{(8.54 \pm 0.07) \times 10^{-1}}$

bathymetric mapping[2]. The workspace dimensions are defined as 31×31 meters. For better visual comparison, the elevation maps of the four environments are shown later in Fig. 7. Thumbnails of these environments are also provided in the second column of Table I for easier reference. These environments have distinct elevation patterns, which helps evaluate the performance of the compared methods across diverse scenarios, offering insights into their generalization capabilities.

- **Env1**: The topography is relatively smooth on the left side and the center, but rocky in the other three directions.
- **Env2**: The terrain is relatively flat in the left two-thirds and mountainous in the remaining one-third.
- **Env3**: There is high variability in the lower part and small variation in the upper section.
- **Env4**: It has two rugged regions at the top and bottom, and a flat area in the upper-left corner.

Compared Methods As discussed in Section IV-A, optimizing the locations of inducing points while learning an input-dependent lengthscale in the AK via gradient optimization is challenging. Consequently, the vanilla Streaming Sparse GP (SSGP) [6] and the original Online Variational Conditioning (OVC) [43] methods do not perform well. To make these baseline methods comparable, we apply the same PCD strategy for selecting the inducing inputs for SSGP and OVC, without further gradient optimization. These improved baselines are referred to as SSGP++ and OVC++. All methods utilize 500 inducing points and are optimized with 10 gradient steps per decision epoch. We keep all other settings unchanged, modifying only the update strategies for the inducing inputs, variational parameters, and hyperparameters.

Evaluation Metrics Following standard practice in the GP literature [55], we use Standardized Mean Squared Error (SMSE) and Mean Standardized Log Loss (MSLL) to evaluate model accuracy and uncertainty quantification. SMSE is calculated as the mean squared error divided by the variance of the test targets. After standardization, a simple method that predicts using the mean of the training targets will have an SMSE of approximately 1. MSLL standardizes the log loss by subtracting the log loss of a naive model that predicts using a Gaussian distribution with the mean and variance of the training targets. An MSLL value close to zero indicates a naive method, while negative values indicate better performance. Additionally, we compare the training time to assess computational efficiency. Since all methods are variants of SVGP, prediction times are similar and thus not included in the comparison.

B. Benchmarking Results

Table I shows that POAM consistently outperforms the baselines in terms of averaged accuracy (SMSE), uncertainty quantification (MSLL), and computational efficiency (training time) in the first three environments. In the fourth environment, OVC achieves a better SMSE than POAM, while POAM still excels in MSLL and training time. The superior SMSE of OVC in this environment is due to its joint optimization of the inducing inputs and lengthscale via stochastic gradient, resulting in a constant lengthscale across the spatial domain. This causes GPR to assign high uncertainty to data-scarce areas, disregarding the complexity of the underlying function, and guiding the robot to explore the environment uniformly,

[2]The elevation map can be viewed and downloaded from the 30-Meter SRTM Tile Downloader https://dwtkns.com/srtm30m/.

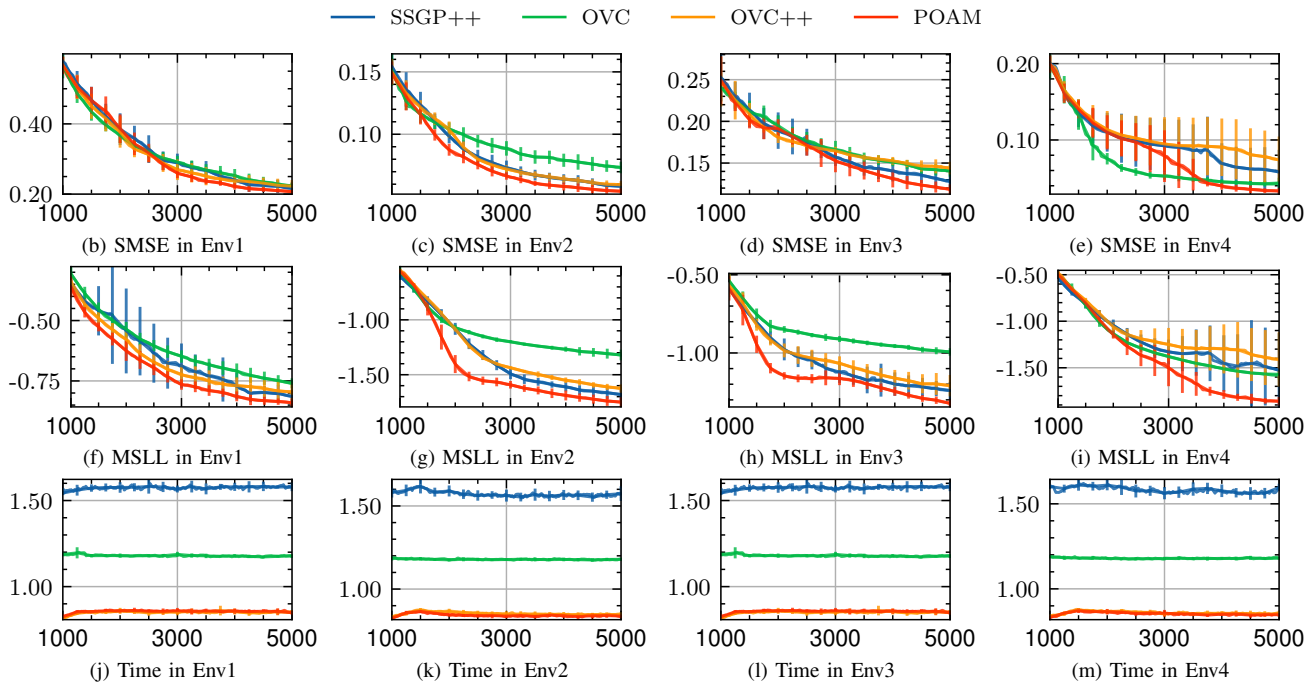


Fig. 5. **Benchmarking results of the evaluated methods across four environments and three metrics**. The proposed method (POAM) is compared against an online sparse GP baseline (`OVC`) and two improved baselines (`OVC++` and `SSGP++`) in terms of standardized mean squared error (SMSE), mean standardized log loss (MSLL), and training time.

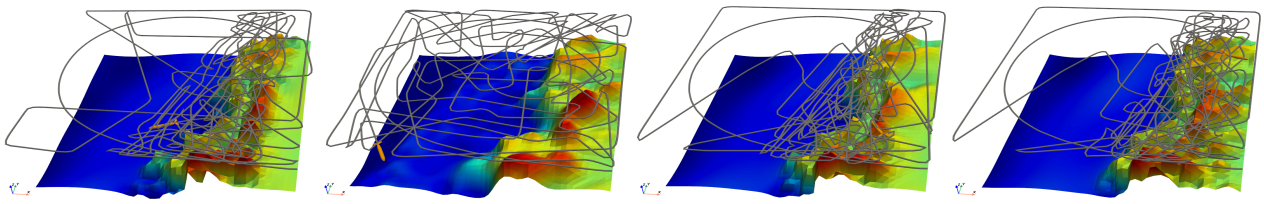
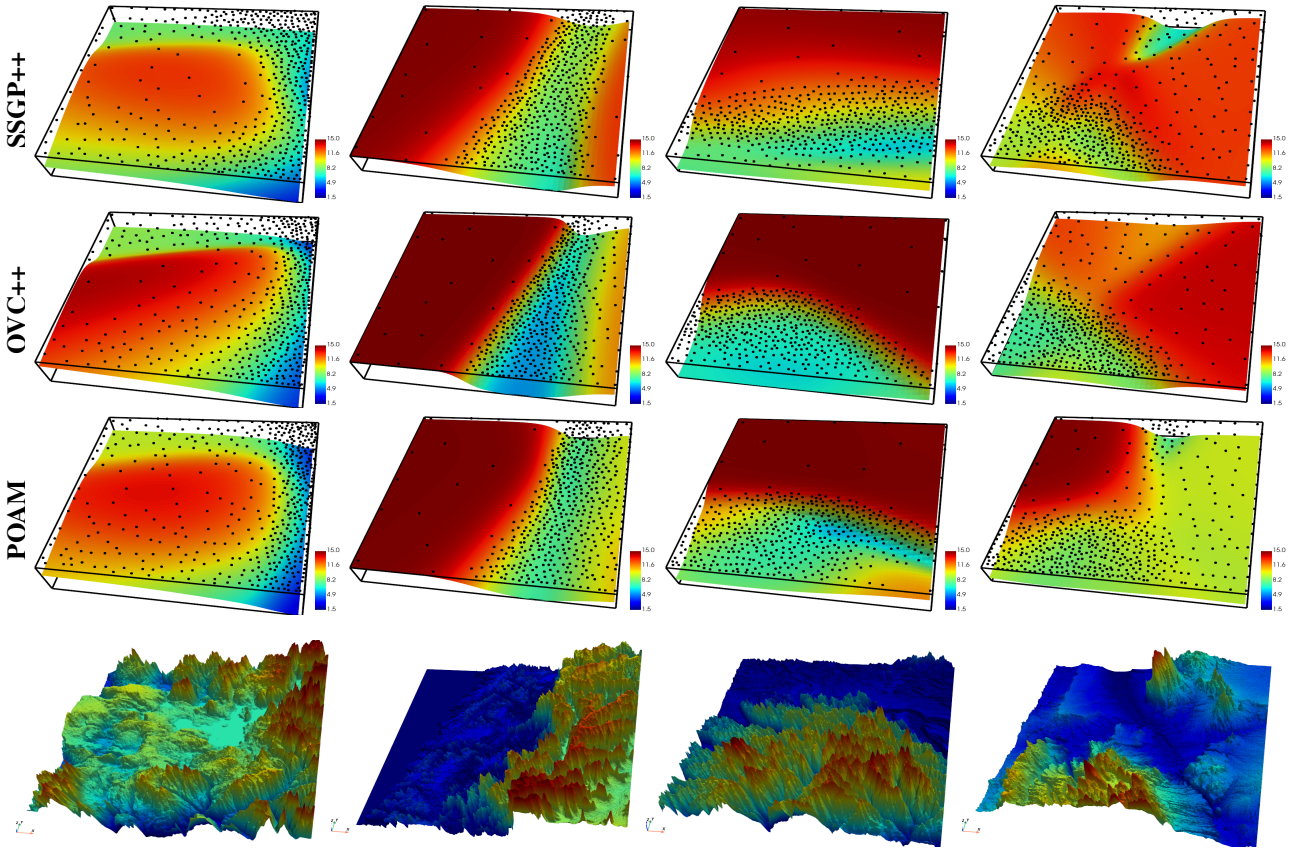


Fig. 6. **Prediction and sampling path of different methods in `Env2`**. The sampling paths of `SSGP++` and `OVC++` do not cover the right boundary extensively enough. `OVC` uniformly explores the environment. `POAM` effectively discovers and covers the complex region.

thus increasing the likelihood of discovering the two isolated rugged regions. In contrast, the other three methods learn input-dependent lengthscales. After discovering one rugged region, these models assign high uncertainty to the complex region, causing the robot to stay and collect more samples until the uncertainty in that area is significantly reduced. This delayed discovery of the other rugged region results in a slower decrease in SMSE.

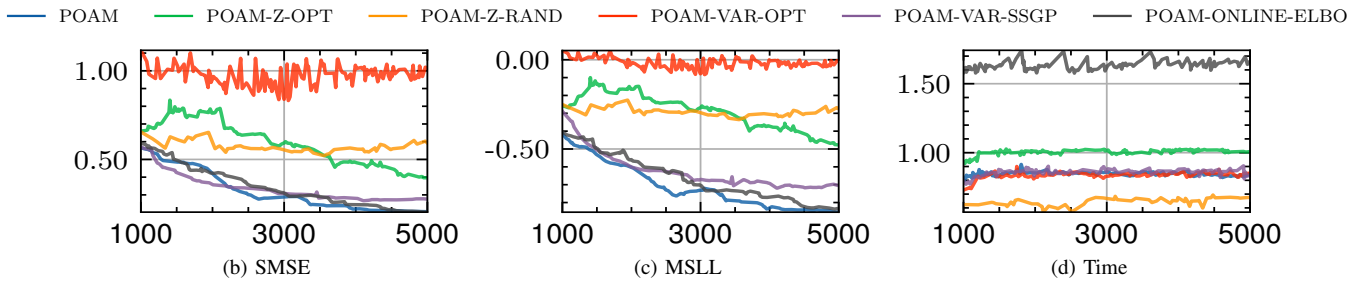
Fig. 5 shows the changes in SMSE, MSLL, and training time as the robot collects between 1000 and 5000 samples. POAM exhibits the most rapid decline in both SMSE and MSLL across all environments except the fourth. In Env4, the SMSE curve for POAM initially shows a higher SMSE than OVC and decreases more slowly. However, after collecting around 3000 samples, POAM quickly catches up and surpasses OVC when sampling in the second rugged region. Thus while Table I shows that, when averaging over decision epochs, the SMSE of OVC in Env4 is lower, the final SMSE value of POAM is better. SSGP++, OVC++, and POAM have higher standard deviations in SMSE and MSLL curves because their performance depends on when the robot encounters rugged regions. Notably, POAM's MSLL curve is significantly lower than those of other methods, indicating better uncertainty quantification. The myopic planner used in the experiments focuses only on a nearby waypoint with the highest entropy, making it challenging to guide the robot to other important regions after discovering the first rugged region unless the uncertainty in the first region is significantly reduced. An advanced informative planner could better utilize POAM's uncertainty quantification capability to improve performance.

The training time curves consistently demonstrate constant runtimes across all methods. POAM and OVC++ have the fastest runtimes, closely followed by OVC. SSGP++ has the slowest runtime due to the complex online ELBO computation. The vanilla SSGP can be more time-consuming because it optimizes all parameters until convergence, whereas SSGP++



(a) Env1 (b) Env2 (c) Env3 (d) Env4

Fig. 7. **Visualization of the four environments and the learned lengthscale maps**. Red means high elevation while blue indicates low. The black dots represent the inducing points.



(b) SMSE (c) MSLL (d) Time

Fig. 8. **Results of ablation study**. The experiment is conducted in Env1. Changing the inducing input update strategy to gradient-based optimization (`POAM-Z-OPT`) or random allocation (`POAM-Z-RAND`) deteriorates the performance. Updating the variational parameters using SSGP's strategy (`POAM-VAR-SSGP`) slightly reduces the performance. Training with the online evidence lower bound of SSGP (`POAM-ONLINE-ELBO`) does not change SMSE or MSLL but increases training time.

only takes several gradient steps. OVC++ has a runtime nearly identical to POAM, as these methods are similar, differing only in what they cache. POAM saves variables related to the training data, while OVC++ saves *noise-weighted* data-dependent variables. This subtle difference significantly enhances POAM's accuracy and uncertainty quantification capabilities.

The most substantial performance disparity among the four environments is evident in Env2, which is therefore used for a qualitative assessment. Fig. 6 shows the prediction and sampling trajectories of the four methods. Notably, OVC demonstrates uniform exploration throughout the environment because it faces challenges in learning an input-dependent length scale for AK. In contrast, the other three methods guide the robot to collect more samples in the mountainous region. Additionally, SSGP++ and OVC++ ignore the complex plateau at the right border and primarily gather samples along the steep slope. POAM captures the plateau comprehensively, collecting samples in a more balanced manner.

Fig. 7 presents the learned lengthscale and inducing inputs

across all environments. OVC's lengthscale maps are excluded because they remain flat in all environments. Comparing the lengthscale maps in Env1 to those learned with dense full data in Fig. 3b, we observe that all three methods appropriately learn the input-dependent lengthscale from sequentially received data batches.

While the lengthscale maps appear similar in Env1, the learned lengthscale maps differ in the other three environments. In Env2, SSGP++ and OVC++ show a large lengthscale in the far-right region, whereas POAM assigns a smaller lengthscale to the same area. This results in SSGP++ and OVC++ allocating low uncertainty and fewer inducing inputs in this region, leading to sparser sampling near the right border. In Env3, all three methods assign small length scales in the lower part where the terrain variation is evident. However, POAM delineates the boundary between the two regions more clearly. The most significant differences in lengthscale maps are observed in Env4. While all methods capture the rugged region at the bottom, OVC++ does not place a small lengthscale in the upper rugged region. Additionally, SSGP++ and OVC++ consider the right part of the environment to be smooth, while POAM considers the upper-left corner to be smooth.

C. Ablation Study

To evaluate the effectiveness of each component in our method, we conducted an ablation study by examining various configurations:

- `POAM-Z-OPT`: Optimizes inducing inputs using gradients instead of PCD.
- `POAM-Z-RAND`: Selects inducing inputs randomly rather than using PCD.
- `POAM-VAR-OPT`: Optimizes variational parameters using gradients instead of computing them analytically.
- `POAM-VAR-SSGP`: Replaces the variational parameter update rules of POAM with those of SSGP.
- `POAM-ONLINE-ELBO`: Trains hyperparameters using the online ELBO of SSGP rather than the standard ELBO.

Fig. 8 shows the results of the ablation study. Changing any part of POAM leads to a noticeable drop in performance, except for `POAM-ONLINE-ELBO`, which performs similarly to POAM but is more computationally expensive. The results for `POAM-Z-OPT` and `POAM-VAR-OPT` show that using analytical methods for inducing inputs and variational parameters is better than using gradient-based optimization. Compared to `POAM-Z-RAND`, we see that pivoted Cholesky decomposition is essential for POAM's performance. The result of `POAM-VAR-SSGP` indicates that POAM's method for updating variational parameters is more effective than the SSGP method.

D. Robustness

The appendix include additional experiments that explore the robustness of the algorithm and experimental setting. In particular, we repeat the experiments with a different initialization strategy that uses random points instead of the Bézier curve, and with an uninformed planner that chooses waypoints randomly. The experiments show that the conclusions given above are valid and hence robust to such modifications. In addition, we include a baseline that uses the full dataset for updates of variational parameters. The results show that the performance of POAM is very close to that of using the full dataset, indicating it provides an effective approximation method.

VI. Conclusion

In this paper, we developed the Probabilistic Online Attentive Mapping (POAM) framework to achieve computational efficiency and data efficiency simultaneously in robotic information gathering in non-stationary environments. This is achieved by constant-time model updates and variational Expectation-Maximization training of sparse Gaussian process regression with the Attentive Kernel. Extensive experiments in active bathymetric mapping tasks show that POAM outperforms existing online sparse Gaussian Process models in terms of accuracy, uncertainty quantification, and efficiency.

Despite the promising results, POAM has limitations. The recursive update of the inducing inputs may lead to suboptimal performance in streaming time-series data because some old inducing inputs will be removed in order to make room for new ones, which is a common issue in online sparse GPs. The online update expressions for the variational parameters are for regression tasks. Other tasks such as classification requires different update rules. The hyperparameters are updated slowly, which may not be optimal when prompt hyperparameter adaptation is needed. Addressing these limitations provides opportunities for future work. Additionally, POAM can be extended to other active information gathering tasks such as active implicit surface mapping [39], online active dynamics learning [8], and online active perception for locomotion [50].

Acknowledgements

We acknowledge the support of NSF with grant numbers 2006886, 2047169 and 2246261. We appreciate the constructive comments from the anonymous conference reviewers, which have significantly improved this paper. This research was supported in part by Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute.

References

[1] Akash Arora, P. Michael Furlong, Robert Fitch, Salah Sukkarieh, and Terrence Fong. Multi-modal active perception for information gathering in science missions. *Autonomous Robots (AURO)*, 43(7):1827–1853, October 2019.

[2] Shi Bai, Jinkun Wang, Fanfei Chen, and Brendan Englot. Information-theoretic exploration with Bayesian optimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1816–1822, 2016.

[3] Graeme Best, Oliver M Cliff, Timothy Patten, Ramgopal R Mettu, and Robert Fitch. Dec-MCTS: decentralized planning for multi-robot active perception. *The International Journal of Robotics Research (IJRR)*, 38 (2-3):316–337, 2019.

[4] Jonathan Binney, Andreas Krause, and Gaurav S Sukhatme. Optimizing waypoints for monitoring spatiotemporal phenomena. *The International Journal of Robotics Research (IJRR)*, 32(8):873–888, 2013.

[5] Lorenzo Booth and Stefano Carpin. Informative path planning for scalar dynamic reconstruction using coregionalized Gaussian processes and a spatiotemporal kernel. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8112–8119, 2023.

[6] Thang D Bui, Cuong Nguyen, and Richard E Turner. Streaming sparse Gaussian process approximations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

[7] Thang D Bui, Josiah Yan, and Richard E Turner. A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation. *The Journal of Machine Learning Research (JMLR)*, 18 (1):3649–3720, 2017.

[8] Mona Buisson-Fenet, Friedrich Solowjow, and Sebastian Trimpe. Actively learning gaussian process dynamics. In *Learning for dynamics and control*, pages 5–15, 2020.

[9] David R Burt, Carl Edward Rasmussen, and Mark Van Der Wilk. Convergence of sparse variational inference in Gaussian processes regression. *The Journal of Machine Learning Research (JMLR)*, 21(1):5120–5182, 2020.

[10] Nannan Cao, Kian Hsiang Low, and John M Dolan. Multi-robot informative path planning for active sensing of environmental phenomena: A tale of two algorithms. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 7–14, 2013.

[11] Henry Carrillo, Yasir Latif, Maria L Rodriguez-Arevalo, Jose Neira, and Jose A Castellanos. On the monotonicity of optimality criteria during exploration in active SLAM. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1476–1483. IEEE, 2015.

[12] Benjamin Charrow, Gregory Kahn, Sachin Patil, Sikang Liu, Ken Goldberg, Pieter Abbeel, Nathan Michael, and Vijay Kumar. Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping. In *Proceedings of Robotics: Science and Systems*, pages 1–10, 2015.

[13] Weizhe Chen, Roni Khardon, and Lantao Liu. AK: Attentive Kernel for Information Gathering. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 1–16, 2022.

[14] Weizhe Chen, Roni Khardon, and Lantao Liu. Adaptive Robotic Information Gathering via non-stationary Gaussian processes. *The International Journal of Robotics Research (IJRR)*, pages 1–32, 2023.

[15] Lehel Csató and Manfred Opper. Sparse on-line Gaussian processes. *Neural computation*, 14(3):641–668, 2002.

[16] Gianni A Di Caro and Abdul Wahab Ziaullah Yousaf. Multi-robot informative path planning using a leader-follower architecture. In *2021 International Conference on Robotics and Automation (ICRA)*, pages 10045–10051, 2021.

[17] Dayi E Dong, Henry P Berger, and Ian Abraham. Time Optimal Ergodic Search. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

[18] Matthew Dunbabin and Lino Marques. Robots for environmental monitoring: significant advancements and applications. *IEEE Robotics & Automation Magazine (RAM)*, 19(1):24–39, 2012.

[19] Tom G Farr, Paul A Rosen, Edward Caro, Robert Crippen, Riley Duren, Scott Hensley, Michael Kobrick, Mimi Paller, Ernesto Rodriguez, Ladislav Roth, et al. The shuttle radar topography mission. *Reviews of geophysics*, 45(2), 2007.

[20] Isabel M Rayas Fernández, Christopher E Denniston, David A Caron, and Gaurav S Sukhatme. Informative Path Planning to Estimate Quantiles for Environmental Analysis. *IEEE Robotics and Automation Letters*, 7(4): 10280–10287, 2022.

[21] Genevieve Flaspohler, Victoria Preston, Anna P. M. Michel, Yogesh Girdhar, and Nicholas Roy. Information-guided robotic maximum seek-and-sample in partially observable continuous environments. *IEEE Robotics and Automation Letters (RA-L)*, 4(4):3782–3789, 2019.

[22] Yogesh Girdhar, Philippe Giguère, and Gregory Dudek. Autonomous adaptive exploration using realtime online spatiotemporal topic modeling. *The International Journal of Robotics Research (IJRR)*, 33(4):645–657, 2014.

[23] Helmut Harbrecht, Michael Peters, and Reinhold Schneider. On the low-rank approximation by the pivoted Cholesky decomposition. *Applied numerical mathematics*, 62(4):428–440, 2012.

[24] James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian Processes for Big Data. In *Uncertainty in Artificial Intelligence (UAI)*, page 282. Citeseer, 2013.

[25] 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 (JFR)*, 34(8):1427–1449, 2017.

[26] Trong Nghia Hoang, Quang Minh Hoang, and Bryan Kian Hsiang Low. A unifying framework of anytime sparse Gaussian process regression models with stochastic variational inference for big data. In *International Conference on Machine Learning (ICML)*, pages 569–578, 2015.

[27] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research (JMLR)*, 2013.

[28] Geoffrey A Hollinger and Gaurav S Sukhatme. Sampling-based robotic information gathering algo-

rithms. *The International Journal of Robotics Research (IJRR)*, 33(9):1271–1287, 2014.

[29] Geoffrey A Hollinger, Brendan Englot, Franz S Hover, Urbashi Mitra, and Gaurav S Sukhatme. Active planning for underwater inspection and the benefit of adaptivity. *The International Journal of Robotics Research (IJRR)*, 32(1):3–18, 2013.

[30] Maani Ghaffari Jadidi, Jaime Valls Miro, and Gamini Dissanayake. Sampling-based incremental information gathering with applications to robotic exploration and environmental monitoring. *The International Journal of Robotics Research (IJRR)*, 38(6):658–685, 2019.

[31] Kalvik Jakkala and Srinivas Akella. Multi-Robot Informative Path Planning from Regression with Sparse Gaussian Processes. *arXiv preprint arXiv:2309.07050*, 2023.

[32] Dohyun Jang, Jaehyun Yoo, Clark Youngdong Son, Dabin Kim, and H Jin Kim. Multi-robot active sensing and environmental model learning with distributed Gaussian process. *IEEE Robotics and Automation Letters (RA-L)*, 5(4):5905–5912, 2020.

[33] Liren Jin, Julius Rückin, Stefan H Kiss, Teresa Vidal-Calleja, and Marija Popović. Adaptive-resolution field mapping using Gaussian process fusion with integral kernels. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7471–7478, 2022.

[34] Yiannis Kantaros, Brent Schlotfeldt, Nikolay Atanasov, and George J. Pappas. Sampling-based planning for non-myopic multi-robot information gathering. *Autonomous Robots (AURO)*, 2021.

[35] Oussama Khatib, Xiyang Yeh, Gerald Brantner, Brian Soe, Boyeon Kim, Shameek Ganguly, Hannah Stuart, Shiquan Wang, Mark Cutkosky, Aaron Edsinger, et al. Ocean one: A robotic avatar for oceanic discovery. *IEEE Robotics & Automation Magazine*, 23(4):20–29, 2016.

[36] Yves Kompis, Luca Bartolomei, Ruben Mascaro, Lucas Teixeira, and Margarita Chli. Informed sampling exploration path planner for 3d reconstruction of large scenes. *IEEE Robotics and Automation Letters (RA-L)*, 6(4):7893–7900, 2021.

[37] George P Kontoudis and Michael Otte. Adaptive exploration-exploitation active learning of Gaussian processes. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9448–9455. IEEE, 2023.

[38] Alberto Quattrini Li. Exploration and mapping with groups of robots: recent trends. *Current Robotics Reports*, pages 1–11, 2020.

[39] Liyang Liu, Simon Fryc, Lan Wu, Thanh Long Vu, Gavin Paul, and Teresa Vidal-Calleja. Active and interactive mapping with dynamic gaussian process implicit surfaces for mobile manipulators. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):3679–3686, 2021.

[40] Kian Hsiang Low, John Dolan, and Pradeep Khosla. Information-theoretic approach to efficient adaptive path planning for mobile robotic environmental sensing. In *International Conference on Automated Planning and Scheduling (ICAPS)*, volume 19, pages 233–240, 2009.

[41] Wenhao Luo and Katia Sycara. Adaptive sampling and online learning in multi-robot sensor coverage with mixture of gaussian processes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6359–6364, 2018.

[42] Kai-Chieh Ma, Lantao Liu, and Gaurav S Sukhatme. Informative planning and online learning with sparse Gaussian processes. In *International Conference on Robotics and Automation (ICRA)*, pages 4292–4298, 2017.

[43] Wesley J Maddox, Samuel Stanton, and Andrew G Wilson. Conditioning sparse variational gaussian processes for online decision-making. *Advances in Neural Information Processing Systems*, 34:6365–6379, 2021.

[44] Sandeep Manjanna Manjanna, Alberto Quattrini Li, Ryan N. Smith, Ioannis Rekleitis, and Gregory Dudek. Heterogeneous multi-robot system for exploration and strategic water sampling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4873–4880, 2018.

[45] Roman Marchant and Fabio Ramos. Bayesian optimisation for intelligent environmental monitoring. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2242–2249, 2012.

[46] 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, 2019.

[47] Alexandra Meliou, Andreas Krause, Carlos Guestrin, and Joseph M Hellerstein. Nonmyopic informative path planning in spatio-temporal models. In *The AAAI Conference on Artificial Intelligence (AAAI)*, volume 10, pages 16–7, 2007.

[48] Brady Moon, Satrajit Chatterjee, and Sebastian Scherer. Tigris: An informed sampling-based algorithm for informative path planning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5760–5766, 2022.

[49] Philippe Morere, Roman Marchant, and Fabio Ramos. Sequential Bayesian optimization as a POMDP for environment monitoring with UAVs. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6381–6388, 2017.

[50] Kasidit Muenprasitivej, Jesse Jiang, Abdulaziz Shamsah, Samuel Coogan, and Ye Zhao. Bipedal Safe Navigation over Uncertain Rough Terrain: Unifying Terrain Mapping and Locomotion Stability. *arXiv preprint arXiv:2403.16356*, 2024.

[51] Farzad Niroui, Kaicheng Zhang, Zendai Kashino, and Goldie Nejat. Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):610–617, 2019.

[52] Ruofei Ouyang, Kian Hsiang Low, Jie Chen, and Patrick Jaillet. Multi-robot active sensing of non-stationary Gaussian process-based environmental phenomena. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 573–580, 2014.

[53] Marija Popovic, Joshua Ott, Julius Rückin, and Mykel J Kochendorfer. Robotic Learning for Adaptive Informative Path Planning. *arXiv preprint arXiv:2404.06940*, 2024.

[54] Joaquin Quinonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research (JMLR)*, 6:1939–1959, 2005.

[55] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. The MIT Press, 2005.

[56] Zhongqiang Ren, Akshaya Kesarimangalam Srinivasan, Bhaskar Vundurthy, Ian Abraham, and Howie Choset. A pareto-optimal local optimization framework for multiobjective ergodic search. *IEEE Transactions on Robotics (T-RO)*, 2023.

[57] Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J. Pappas. Anytime planning for decentralized multi-robot active information gathering. *IEEE Robotics and Automation Letters (RA-L)*, 3(2): 1025–1032, 2018.

[58] Brent Schlotfeldt, Vasileios Tzoumas, and George J Pappas. Resilient active information acquisition with teams of robots. *IEEE Transactions on Robotics (T-RO)*, 2021.

[59] Lukas Maximilian Schmid, Michael Pantic, Raghav Khanna, Lionel Ott, Roland Siegwart, and Juan Nieto. An efficient sampling-based method for online informative path planning in unknown environments. *IEEE Robotics and Automation Letters (RA-L)*, 2020.

[60] Rishit Sheth, Yuyang Wang, and Roni Khardon. Sparse variational inference for generalized GP models. In *International Conference on Machine Learning (ICML)*, pages 1302–1311, 2015.

[61] Amarjeet Singh, Andreas Krause, Carlos Guestrin, William Kaiser, and Maxim Batalin. Efficient planning of informative paths for multiple robots. In *International Joint Conference on Artifical Intelligence (IJCAI)*, pages 2204–2211, 2007.

[62] Samuel Stanton, Wesley Maddox, Ian Delbridge, and Andrew Gordon Wilson. Kernel interpolation for scalable online Gaussian processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3133–3141, 2021.

[63] Muchen Sun, Ayush Gaggar, Peter Trautman, and Todd Murphey. Fast Ergodic Search with Kernel Functions. *arXiv preprint arXiv:2403.01536*, 2024.

[64] Yoonchang Sung and Pratap Tokekar. A competitive algorithm for online multi-robot exploration of a translating plume. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3391–3397, 2019.

[65] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[66] Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 567–574, 2009.

[67] Yang Xu, Ronghao Zheng, Senlin Zhang, Meiqin Liu, and Shoudong Huang. CARE: Confidence-rich Autonomous Robot Exploration using Bayesian Kernel Inference and Optimization. *IEEE Robotics and Automation Letters*, 2023.

[68] Jingjin Yu, Mac Schwager, and Daniela Rus. Correlated orienteering problem and its application to informative path planning for persistent monitoring tasks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 342–349, 2014.

[69] Jing Zeng, Yanxu Li, Yunlong Ran, Shuo Li, Fei Gao, Lincheng Li, Shibo He, Jiming Chen, and Qi Ye. Efficient view path planning for autonomous implicit reconstruction. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4063–4069, 2023.

[70] Zhengdong Zhang, Theia Henderson, Sertac Karaman, and Vivienne Sze. FSMI: fast computation of Shannon mutual information for information-theoretic mapping. *The International Journal of Robotics Research (IJRR)*, 39(9):1155–1177, 2020.

[71] Hai Zhu, Jen Jen Chung, Nicholas RJ Lawrance, Roland Siegwart, and Javier Alonso-Mora. Online informative path planning for active information gathering of a 3D surface. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1488–1494, 2021.

APPENDIX

The appendix provides a detailed derivation and explanation of prior work and how POAM fits in that context, as well as additional experimental results showing the robustness of POAM and the experimental setup used in the main paper.

A. Detailed Discussion of SSGP and OVC

We provide detailed explanations on the updates of variational parameters and hyperparameters in the related work. Additionally, we describe how we adapt existing methods to serve as robust baselines for our proposed approach. We analyze various methods from the perspective of updating data-dependent variables to highlight their similarities and differences, positioning our work within the existing methods for online sparse Gaussian process regression. Consistent mathematical notations, as summarized in Table II, are used throughout the paper.

1) Sparse Gaussian Process Regression (SGPR): We reintroduce the predictive distribution of SGPR for easier refer-

TABLE II
MATHEMATICAL NOTATIONS.

Meaning	Example	Remark
variable	n	lower-case
constant	N	upper-case
vector	$\mathbf{x}$	bold, lower-case
matrix	$\mathbf{X}$	bold, upper-case
set/space	$\mathbb{R}$	blackboard
function	$f(\cdot)$	
functional	$\mathtt{KL}[\cdot]$	typewriter with square brackets
special density	$\mathcal{N}$	calligraphy capital
definition	$\triangleq$	normal
transpose	$\mathbf{m}^\intercal$	customized command
Euclidean norm	$\|\cdot\|_2$	customized command

ence:

$$
\begin{aligned}
q(f_\star) =& \mathcal{N}\left(f_\star \mid \tilde{\mu}, \tilde{\nu}\right) \\
\tilde{\mu} =& \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{m}, \\
\tilde{\nu} =& k_{\star\star} - \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{k}_{u\star} + \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{S} \mathbf{K}_{uu}^{-1} \mathbf{k}_{u\star}, \\
\mathbf{S} =& \mathbf{K}_{uu} \mathbf{A}^{-1} \mathbf{K}_{uu}, \qquad (22) \\
\mathbf{m} =& \mathbf{K}_{uu} \mathbf{A}^{-1} \mathbf{K}_{uf} \boldsymbol{\Sigma}^{-1} \mathbf{y}, \qquad (23) \\
\mathbf{A} =& \mathbf{K}_{uu} + \mathbf{K}_{uf} \boldsymbol{\Sigma}^{-1} \mathbf{K}_{uf}^\intercal.
\end{aligned}
$$

Here, $\mathbf{k}_{u\star}$ is the kernel vector between the inducing inputs and the test inputs, $\mathbf{K}_{uu}$ is the kernel matrix of the inducing inputs, $\mathbf{K}_{uf}$ is the kernel matrix between the inducing inputs and the training inputs, and $\boldsymbol{\Sigma}$ is a diagonal matrix with the noise variance σ^2 on the diagonal. We discuss different methods from the perspective of how they update the variational parameters $\mathbf{m}$ and $\mathbf{S}$ via the data-dependent terms, which are highlighted in blue (for the $\mathbf{y}$-related term) and red.

2) Streaming Sparse Gaussian Processes (SSGP): Streaming sparse GPs (SSGP) extends SGPR to the streaming setting where data arrives sequentially and the model can only process each batch of data once before discarding it [6]. To derive an online update that facilitates hyperparameter optimization, SSGP approximates the likelihood of the old data with the old posterior $q'(\mathbf{u}')$ and prior $p'(\mathbf{u}')$ at the old inducing inputs $\mathbf{u}'$. This results in two additional KL regularization terms in the ELBO:

$$
\begin{aligned}
\mathtt{ELBO} = & \sum_{n=1}^{N_{\text{new}}} \mathrm{E}_{q(f_n)}\left[\log p(y_n \mid f_n)\right] - \mathtt{KL}[q(\mathbf{u}) \parallel p(\mathbf{u})] \\
& + \mathtt{KL}[q(\mathbf{u}') \parallel p'(\mathbf{u}')] - \mathtt{KL}[q(\mathbf{u}') \parallel q'(\mathbf{u}')]. \qquad (24)
\end{aligned}
$$

The first line of Eq. (24) is the uncollapsed ELBO of SVGP in Eq. (10) applied to the new data, and the second line involves two KL regularization terms for online update.

As in SGPR, this generic ELBO can be further simplified in the regression case to a collapsed form:

$$
\begin{aligned}
\mathtt{ELBO} = & \log \mathcal{N}(\tilde{\mathbf{y}} \mid \mathbf{0}, \mathbf{Q}_{\tilde{y}\tilde{y}}) - \frac{1}{2\sigma^2} \mathrm{tr}(\mathbf{K}_{ff} - \mathbf{Q}_{ff}) \\
& - \frac{1}{2} \mathrm{tr}(\hat{\boldsymbol{\Sigma}}^{-1}(\mathbf{K}_{u'u'} - \mathbf{Q}_{u'u'})) + \text{constants}. \qquad (25)
\end{aligned}
$$

We follow appendix C.4 of Maddox et al. [43] and collect variables that are not related to the trainable parameters to the `constants` term. Here, some notations are introduced to reveal the similarity and difference between the collapsed ELBO of SSGP and that of SGPR:

$$
\begin{aligned}
\hat{\boldsymbol{\Sigma}} &= (\mathbf{S}'^{-1} - \mathbf{K}'^{-1}_{u'u'})^{-1}, & \mathbf{Q}_{u'u'} &= \mathbf{K}_{uu'}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{K}_{uu'}, \\
\tilde{\mathbf{y}} &= [\hat{\mathbf{y}}^\intercal, \mathbf{y}^\intercal]^\intercal, & \hat{\mathbf{y}} &= \hat{\boldsymbol{\Sigma}} \mathbf{S}'^{-1} \mathbf{m}', \\
\mathbf{Q}_{\tilde{y}\tilde{y}} &= \mathbf{Q}_{\tilde{f}\tilde{f}} + \tilde{\boldsymbol{\Sigma}}, & \mathbf{Q}_{\tilde{f}\tilde{f}} &= \mathbf{K}_{u\tilde{f}}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{K}_{u\tilde{f}}, \\
\tilde{\boldsymbol{\Sigma}} &= \begin{bmatrix} \hat{\boldsymbol{\Sigma}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma} \end{bmatrix}, & \mathbf{K}_{u\tilde{f}} &= [\mathbf{K}_{uu'}, \mathbf{K}_{uf}]. \qquad (26)
\end{aligned}
$$

The first line of Eq. (25) has a similar form as the collapsed ELBO of SGPR in Eq. (6) and the second trace term regularizes the new inducing inputs to be close to the old ones. Other terms that are not related to the trainable parameters are represented as "constants" in the equation above. With this online ELBO, the old data is not needed and hyperparameters and inducing inputs can be directly optimized using only new data. To aid the optimization of the inducing inputs, Bui et al. [6] also propose a resampling heuristic for initialization of inducing inputs. Specifically, some randomly selected old inducing inputs are moved to randomly selected inputs in the new batch.

From Eq. (42) and Eq. (45) in Bui et al. [6]'s appendix[3], the predictive distribution is given by:

$$
\begin{aligned}
q(f_\star) =& \mathcal{N}\left(f_\star \mid \tilde{\mu}, \tilde{\nu}\right), \\
\tilde{\mu} =& \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \tilde{\mathbf{m}}, \\
\tilde{\nu} =& k_{\star\star} - \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \mathbf{k}_{u\star} + \mathbf{k}_{u\star}^\intercal \mathbf{K}_{uu}^{-1} \tilde{\mathbf{S}} \mathbf{K}_{uu}^{-1} \mathbf{k}_{u\star}, \\
\text{where } \tilde{\mathbf{S}} =& (\mathbf{K}_{uu}^{-1} + \mathbf{K}_{uu}^{-1} \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \mathbf{K}_{\tilde{f}u} \mathbf{K}_{uu}^{-1})^{-1}, \\
\tilde{\mathbf{m}} =& \tilde{\mathbf{S}} \mathbf{K}_{uu}^{-1} \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\mathbf{y}}.
\end{aligned}
$$

To see how the data-dependent terms are updated, we simplify the expressions of $\tilde{\mathbf{S}}$ to make it similar to SGPR's expression in Eq. (22):

$$
\begin{aligned}
\tilde{\mathbf{S}} =& (\mathbf{K}_{uu}^{-1} + \mathbf{K}_{uu}^{-1} \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \mathbf{K}_{\tilde{f}u} \mathbf{K}_{uu}^{-1})^{-1} \\
=& \mathbf{K}_{uu} \mathbf{K}_{uu}^{-1} (\mathbf{K}_{uu}^{-1} + \mathbf{K}_{uu}^{-1} \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \mathbf{K}_{\tilde{f}u} \mathbf{K}_{uu}^{-1})^{-1} \mathbf{K}_{uu}^{-1} \mathbf{K}_{uu} \\
=& \mathbf{K}_{uu} \underbrace{(\mathbf{K}_{uu} + \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \mathbf{K}_{\tilde{f}u})^{-1}}_{\tilde{\mathbf{A}}} \mathbf{K}_{uu}. \qquad (27)
\end{aligned}
$$

Plugging the simplified $\tilde{\mathbf{S}}$ into $\tilde{\mathbf{m}}$, we have an $\tilde{\mathbf{m}}$ expression that resembles SGPR's $\mathbf{m}$ in Eq. (23):

$$
\tilde{\mathbf{m}} = \mathbf{K}_{uu} \tilde{\mathbf{A}}^{-1} \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\mathbf{y}}. \qquad (28)
$$

According to Eq. (33) and Eq. (38) in Bui et al. [6]'s appendix, these data-dependent terms are expanded as

$$
\begin{aligned}
& \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\mathbf{y}} \\
=& \mathbf{K}_{uf} \boldsymbol{\Sigma}^{-1} \mathbf{y} + \mathbf{K}_{u'u}^\intercal \mathbf{S}'^{-1} \mathbf{m}', \qquad (29) \\
& \mathbf{K}_{u\tilde{f}} \tilde{\boldsymbol{\Sigma}}^{-1} \mathbf{K}_{\tilde{f}u} \\
=& \mathbf{K}_{uf} \boldsymbol{\Sigma}^{-1} \mathbf{K}_{fu} + \mathbf{K}_{u'u}^\intercal (\mathbf{S}'^{-1} - \mathbf{K}'^{-1}_{u'u'}) \mathbf{K}_{u'u}. \qquad (30)
\end{aligned}
$$

[3]We refer to equations in their revised arXiv paper available at https://arxiv.org/abs/1705.07131, noting that the equation numbering differs from the proceeding version.

Now we can see that SSGP updates the variational parameters $\mathbf{m}$ and $\mathbf{S}$ by saving the old variational parameters and kernel matrix (highlighted in green), correcting/projecting them via the cross-covariance matrix $\mathbf{K}_{u'u}$, and adding the new data-dependent terms $\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{y}$ and $\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{K}_{fu}$.

3) Online Variational Conditioning (OVC): Observing the similarity between Eqs. (22) and (27) as well as Eqs. (23) and (28), Maddox et al. [43] view SSGP's variational update as a sequence of SGPR updates on an *augmented* dataset $\{\tilde{\mathbf{X}}, \tilde{\mathbf{y}}\}$ that consists of pseudo data $\{\mathbf{Z}', \hat{\mathbf{y}}\}$ and the current data $\{\mathbf{X}, \mathbf{y}\}$ through a special likelihood function $p(\tilde{\mathbf{y}} \,|\, \tilde{\mathbf{f}}) = \mathcal{N}(\tilde{\mathbf{y}} \mid \mathbf{0}, \tilde{\boldsymbol{\Sigma}})$. Specifically, the augmented covariance matrix $\tilde{\boldsymbol{\Sigma}}$ is a block-diagonal matrix with the pseudo covariance $\hat{\boldsymbol{\Sigma}}$ and the new covariance $\boldsymbol{\Sigma}$ on the diagonal, and the augmented dataset is defined as $\tilde{\mathbf{X}} = [\mathbf{Z}'^\intercal, \mathbf{X}^\intercal]^\intercal, \tilde{\mathbf{y}} = [\hat{\mathbf{y}}^\intercal, \mathbf{y}^\intercal]^\intercal$. Intuitively, the information of the old data is preserved in the pseudo data $\{\mathbf{Z}', \hat{\mathbf{y}}\}$ and the pseudo covariance $\hat{\boldsymbol{\Sigma}}$. In the following, variables related to the augmented dataset are denoted with a tilde symbol $\tilde{\cdot}$ while those related to the pseudo dataset are denoted with a hat $\hat{\cdot}$.

Computing the pseudo targets $\hat{\mathbf{y}}$ and the pseudo covariance matrix $\hat{\boldsymbol{\Sigma}}$ is the key step in constructing the augmented dataset. Maddox et al. [43] show that (*cf.* Eq. (A.6) and Eq. (A.7)) they can be derived by reversing the equations of the old variational parameters $\mathbf{m}'$ and $\mathbf{S}'$ in Eqs. (22) and (23) when assuming that the old data are observed only at the old inducing inputs $\mathbf{X}' = \mathbf{Z}'$:

$$
\begin{aligned}
&\mathbf{S}' = \mathbf{K}'_{u'u'}(\mathbf{K}'_{u'u'} + \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}'_{u'u'})^{-1}\mathbf{K}'_{u'u'} \\
\Leftrightarrow &\mathbf{K}'^{-1}_{u'u'}\mathbf{S}'\mathbf{K}'^{-1}_{u'u'} = (\mathbf{K}'_{u'u'} + \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}'_{u'u'})^{-1} \\
\Leftrightarrow &\mathbf{K}'_{u'u'}\mathbf{S}'^{-1}\mathbf{K}'_{u'u'} = \mathbf{K}'_{u'u'} + \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}'_{u'u'} \\
\Leftrightarrow &\mathbf{K}'_{u'u'}\mathbf{S}'^{-1}\mathbf{K}'_{u'u'} - \mathbf{K}'_{u'u'} = \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}'_{u'u'} \\
\Leftrightarrow &\mathbf{S}'^{-1} - \mathbf{K}'^{-1}_{u'u'} = \hat{\boldsymbol{\Sigma}}^{-1} \\
\Leftrightarrow &\hat{\boldsymbol{\Sigma}} = (\mathbf{S}'^{-1} - \mathbf{K}'^{-1}_{u'u'})^{-1}, \qquad (31) \\
&\mathbf{m}' = \mathbf{K}'_{u'u'}(\mathbf{K}'_{u'u'} + \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}'_{u'u'})^{-1}\mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\hat{\mathbf{y}} \\
\Leftrightarrow &\hat{\mathbf{y}} = \hat{\boldsymbol{\Sigma}}\mathbf{K}'^{-1}_{u'u'}(\mathbf{K}'_{u'u'} + \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}'_{u'u'})\mathbf{K}'^{-1}_{u'u'}\mathbf{m}' \\
\Leftrightarrow &\hat{\mathbf{y}} = \hat{\boldsymbol{\Sigma}}(\mathbf{K}'^{-1}_{u'u'} + \hat{\boldsymbol{\Sigma}}^{-1})\mathbf{m}' \\
\Leftrightarrow &\hat{\mathbf{y}} = \hat{\boldsymbol{\Sigma}}(\mathbf{K}'^{-1}_{u'u'} + \mathbf{S}'^{-1} - \mathbf{K}'^{-1}_{u'u'})\mathbf{m}' = \hat{\boldsymbol{\Sigma}}\mathbf{S}'^{-1}\mathbf{m}'. \qquad (32)
\end{aligned}
$$

The expressions of $\hat{\boldsymbol{\Sigma}}$ and $\hat{\mathbf{y}}$ are consistent with those defined in Eq. (26) and substituting the augmented dataset and covariance into the $\mathbf{m}$ and $\mathbf{S}$ equations of SGPR in Eqs. (22) and (23) yields the same update equations of the variational parameters $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{S}}$ of SSGP.

Following the idea of constructing the pseudo data and covariance matrix by reversing some equations that involve $\hat{\mathbf{y}}, \hat{\boldsymbol{\Sigma}}$ and some saved parameters, Maddox et al. [43] propose to cache the data-dependent terms $\mathbf{c} = \mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{y}$ and $\mathbf{C} = \mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{K}_{uf}^\intercal$, and reverse these equations to obtain the pseudo targets $\hat{\mathbf{y}}$ and pseudo covariance matrix $\hat{\boldsymbol{\Sigma}}$, under the same assumption that the old inputs are the old inducing inputs:

$$
\begin{aligned}
&\mathbf{C}' = \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}'_{u'u'} \\
\Leftrightarrow &\hat{\boldsymbol{\Sigma}}^{-1} = \mathbf{K}'^{-1}_{u'u'}\mathbf{C}'\mathbf{K}'^{-1}_{u'u'}, \qquad (33) \\
&\mathbf{c}' = \mathbf{K}'_{u'u'}\hat{\boldsymbol{\Sigma}}^{-1}\hat{\mathbf{y}} \\
\Leftrightarrow &\hat{\mathbf{y}} = \hat{\boldsymbol{\Sigma}}\mathbf{K}'^{-1}_{u'u'}\mathbf{c}' \\
\Leftrightarrow &\hat{\mathbf{y}} = (\mathbf{K}'^{-1}_{u'u'}\mathbf{C}'\mathbf{K}'^{-1}_{u'u'})^{-1}\mathbf{K}'^{-1}_{u'u'}\mathbf{c}' \\
\Leftrightarrow &\hat{\mathbf{y}} = \mathbf{K}'_{u'u'}\mathbf{C}'^{-1}\mathbf{c}'. \qquad (34)
\end{aligned}
$$

Comparing the expressions of $\hat{\boldsymbol{\Sigma}}$ and $\hat{\mathbf{y}}$ in Eqs. (33) and (34) with those in Eqs. (31) and (32), reversing the equations of data-dependent terms rather than the variational parameters eliminates the subtraction of two inverse matrices in the expression of $\hat{\boldsymbol{\Sigma}}$, which helps numerical stability.

Maddox et al. [43] propose Online Variational Conditioning (OVC) that computes the pseudo covariance matrix and pseudo targets following Eqs. (33) and (34) and feeds the augmented dataset $\{\tilde{\mathbf{X}}, \tilde{\mathbf{y}}\}$ with a pseudo likelihood function $p(\tilde{\mathbf{y}} \,|\, \tilde{\mathbf{f}}) = \mathcal{N}(\tilde{\mathbf{y}} \mid \mathbf{0}, \tilde{\boldsymbol{\Sigma}})$ to a GPR or SGPR for online conditioning – online update of the posterior distribution to condition on the new data. This explains their update of variational parameters. For inducing inputs, they use PCD on the augmented dataset. The target application in OVC is Bayesian optimization, where online conditioning is crucial for computing some advanced look-ahead acquisition functions while hyperparameter optimization is less important, hence not extensively discussed in their paper.

When applying the augmented dataset constructed by OVC to a SGPR, the data-dependent terms of the variational parameters are updated as

$$
\begin{aligned}
&\mathbf{K}_{u\tilde{f}}\tilde{\boldsymbol{\Sigma}}^{-1}\mathbf{K}_{\tilde{f}u} \\
=&\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{K}_{fu} + \mathbf{K}_{uu'}\hat{\boldsymbol{\Sigma}}^{-1}\mathbf{K}_{u'u}, \\
=&\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{K}_{fu} + \mathbf{K}_{u'u}^\intercal(\mathbf{K}'^{-1}_{u'u'}\mathbf{C}'\mathbf{K}'^{-1}_{u'u'})\mathbf{K}_{u'u}, \qquad (35) \\
&\mathbf{K}_{u\tilde{f}}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\mathbf{y}} \\
=&\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{y} + \mathbf{K}_{uu'}\hat{\boldsymbol{\Sigma}}^{-1}\hat{\mathbf{y}}, \\
=&\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{y} + \mathbf{K}_{u'u}^\intercal(\mathbf{K}'^{-1}_{u'u'}\mathbf{C}'\mathbf{K}'^{-1}_{u'u'})\mathbf{K}'_{u'u'}\mathbf{C}'^{-1}\mathbf{c}', \\
=&\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{y} + \mathbf{K}_{u'u}^\intercal\mathbf{K}'^{-1}_{u'u'}\mathbf{c}'. \qquad (36)
\end{aligned}
$$

Eqs. (35) and (36) can be viewed as updating $\mathbf{C}'$ and $\mathbf{c}'$ via a projection matrix $\mathbf{P} \triangleq \mathbf{K}'^{-1}_{u'u'}\mathbf{K}_{u'u}$ and adding the new data-dependent terms:

$$
\begin{aligned}
\mathbf{C} =&\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{K}_{fu} + \mathbf{P}^\intercal\mathbf{C}'\mathbf{P}, \qquad (37) \\
\mathbf{c} =&\mathbf{K}_{uf}\boldsymbol{\Sigma}^{-1}\mathbf{y} + \mathbf{P}^\intercal\mathbf{c}'. \qquad (38)
\end{aligned}
$$

Our proposed POAM shares the same idea of projecting the old data-dependent terms and adding the new data-dependent terms, but we save $\mathbf{K}'_{u'f'}\mathbf{y}'$ and $\mathbf{K}'_{u'f'}\mathbf{K}'_{f'u'}$ instead of $\mathbf{c}'$ and $\mathbf{C}'$, which leads to better performance in our experiments. In other words, the observational noise $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$ is not data-dependent, so we should use the new $\boldsymbol{\Sigma}$ rather than projecting the old one.

TABLE III

SUMMARY OF RELATED WORK AND BASELINE METHODS. SSGP AND OVC ARE TWO EXISTING METHODS THAT ARE CLOSELY RELATED TO POAM, WHICH ARE IMPROVED (I.E., SSGP++ AND OVC++) TO WORK BETTER WITH NON-STATIONARY KERNELS AND SERVE AS STRONG BASELINES.

Method		Saved Variables	Projection Matrix	Hyperparameters Update	Inducing Inputs Update
SSGP [6]	$\mathbf{S'}^{-1}\mathbf{m'}$,	$\mathbf{S'}^{-1} - \mathbf{K'}^{-1}_{u'u'}$	$\mathbf{K}_{u'u}$	L-BFGS-B with Online ELBO	Resampling & Optimization
SSGP++	$\mathbf{S'}^{-1}\mathbf{m'}$,	$\mathbf{S'}^{-1} - \mathbf{K'}^{-1}_{u'u'}$	$\mathbf{K}_{u'u}$	Adam Optimizer with Online ELBO	Pivoted Cholesky Decomposition (PCD)
OVC [43]	$\mathbf{K}_{uf}\mathbf{\Sigma}^{-1}\mathbf{y}$,	$\mathbf{K}_{uf}\mathbf{\Sigma}^{-1}\mathbf{K}_{uf}^{\intercal}$	$\mathbf{K'}^{-1}_{u'u'}\mathbf{K}_{u'u}$		PCD & Optimization
OVC++	$\mathbf{K}_{uf}\mathbf{\Sigma}^{-1}\mathbf{y}$,	$\mathbf{K}_{uf}\mathbf{\Sigma}^{-1}\mathbf{K}_{uf}^{\intercal}$	$\mathbf{K'}^{-1}_{u'u'}\mathbf{K}_{u'u}$	Variational EM, Mini-Batch SGD, ELBO	PCD
POAM (ours)	$\mathbf{K'}_{u'f'}\mathbf{y'}$,	$\mathbf{K'}_{u'f'}\mathbf{K'}_{f'u'}$	$\mathbf{K'}^{-1}_{u'u'}\mathbf{K}_{u'u}$	Variational EM, Mini-Batch SGD, ELBO	PCD

4) Baseline Methods: We found that optimizing the inducing inputs with the online ELBO of SSGP performs poorly in our experiments, regardless of the initialization strategy, so SSGP++ uses pivoted Cholesky decomposition (PCD) [9] for updating inducing inputs and does not optimize them with the online ELBO. Also, SSGP++ uses the Adam optimizer for updating the hyperparameters because the L-BFGS-B optimizer is not suitable for the Attentive Kernel (AK) that has many more parameters than the commonly used stationary kernels. The hyperparameter optimization of OVC is not properly discussed in [43], so we use the same strategy as POAM. The vanilla OVC still optimizes the inducing inputs after initializing them with PCD, while OVC++ does not optimize them. Table III summarizes the differences of each component in the aforementioned methods.

We note that during the experiments, SSGP++ sometimes throws a numerical error, requiring an increase in the "jitter" value added to the diagonal elements of positive-definite matrices for stable Cholesky decomposition and re-running the experiment. Although not reported in the results, this is an important factor to consider when deploying SSGP++ in practice.

B. Additional Experiments

To gauge the robustness of the proposed method to different initial conditions and to disentangle the contributions of the proposed model and the planner, we have conducted two additional sets of experiments. These experiments also include an additional baseline as follows.

- **Random Initialization**: A set of experiments with initial samples and the robot's starting positions *uniformly and randomly* sampled from the environment and the same *max-entropy planner* used in the original experiments. These experiments are conducted to evaluate the robustness of the proposed method to different initial conditions.
- **Random Planner**: A set of experiments with the same random-initialization strategy and a *random planner* that uniformly samples a waypoint at random from the environment at each decision epoch. These experiments compare different models when the robot executes the same path that is independent of the model being evaluated.
- **Full dataset baseline**: We include an additional baseline using the full-update version of POAM (`FULL`) to verify the effectiveness of the proposed online update rule. The `FULL` method uses the full dataset to update the variational parameters instead of using the proposed online update rule, so it is expected to be slower but has better performance compared to `POAM`.

The experiments are repeated 10 times with different random seeds and the mean and standard deviation of the results are reported in Fig. 9.

Results of Random Initialization Experiments Under the randomized initialization, the rankings of the compared methods are consistent across different environments: the proposed method (`POAM`) and its full-update version (`FULL`) outperform the baselines in terms of SMSE and MSLL, `SGGP++` and `OVC++` perform similarly, and `OVC` performs the worst. Since the SMSE of OVC is much worse than other approaches in the random initialization experiments, we use a separate y-axis at the right side of the SMSE plots to better visualize the performance of the other methods.

Results of Random Planner Experiments `POAM` outperforms the baselines in terms of SMSE and MSLL, `SGGP++` and `OVC++` perform similarly, and OVC performs the worst. Additionally, by contrasting the two sets of experiments which only differ in the planner, we can see the performance gain brought by the active learning strategy is also evident.

Comparison with Full Update The difference between the proposed online update rule and the full update is negligible. This indicates that the proposed online update rule is accurate and robust, without significant error accumulation over time.

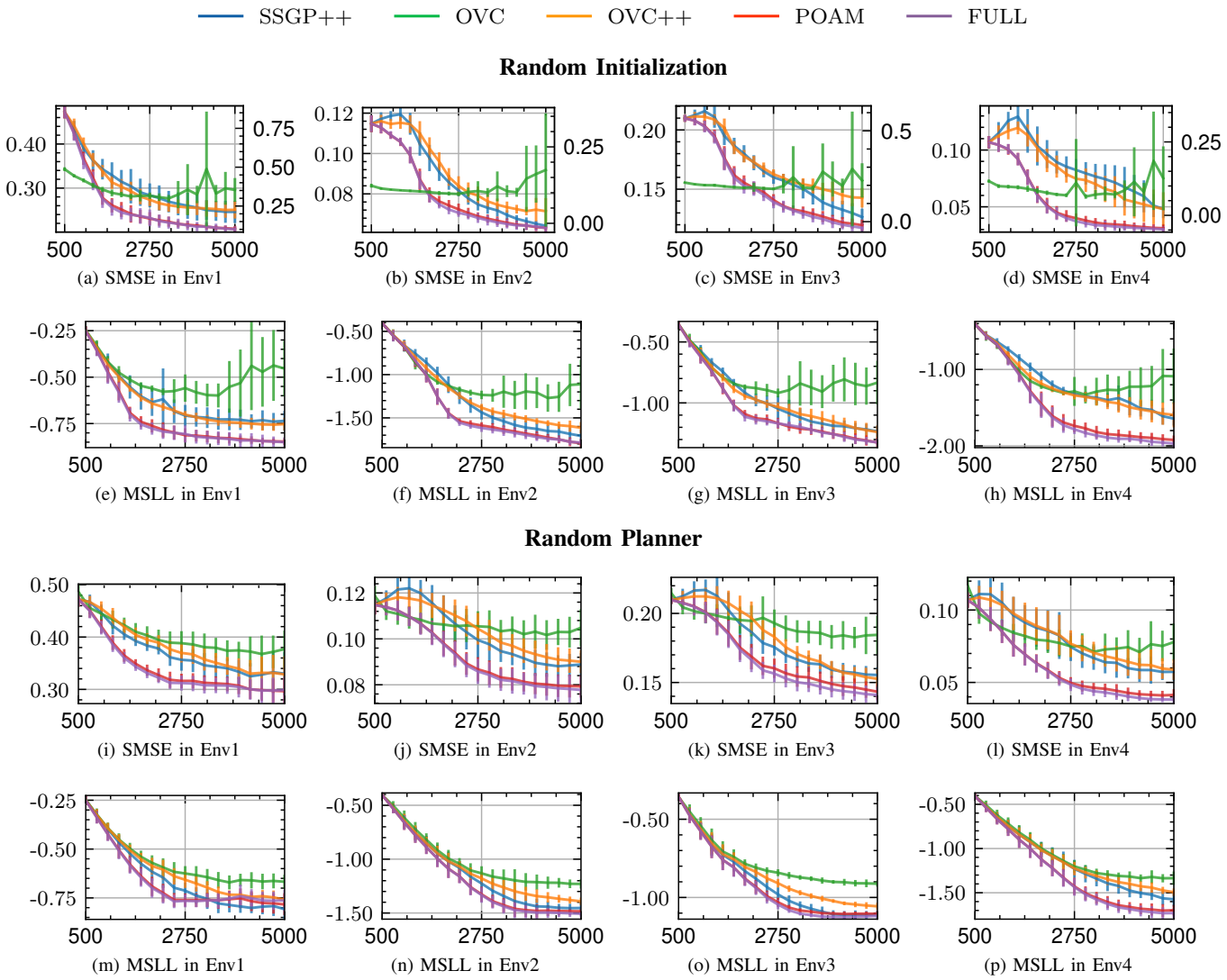


Fig. 9. Standardized Mean Squared Error (SMSE) and Mean standardized log-loss (MSLL) curves for three baseline methods (SSGP++, OVC, OVC++) are compared against the proposed method (POAM) and its full-update version (FULL) which performs a full update of the variational parameters at each time step. The random initialization experiments use the max-entropy planner and the initial samples and robot's starting positions are uniformly and randomly sampled from the environment. The random planner experiments use the same random-initialization strategy and a random planner that uniformly samples a waypoint at random from the environment at each decision epoch. Note that, in the random initialization experiments, the y-axis of SMSE on the right is for the OVC method alone, as its performance is much worse than the other approaches.