AK: Attentive Kernel for Information Gathering

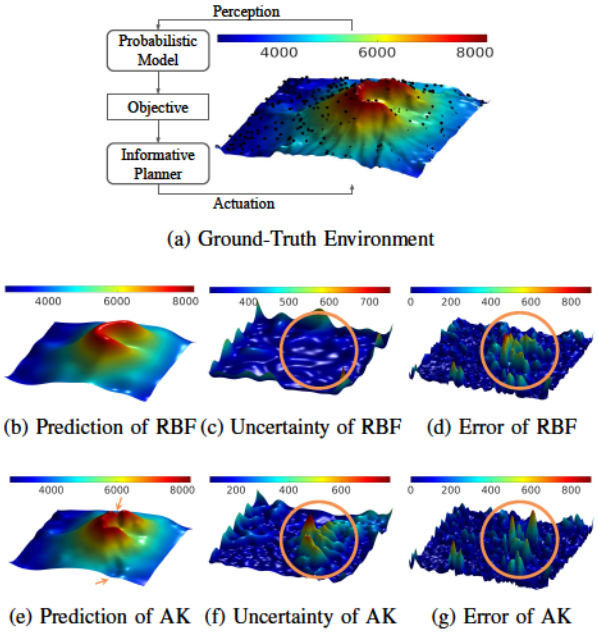
Weizhe Chen, Roni Khardon and Lantao Liu
Indiana University, Bloomington, IN, USA, 47408
Emails: {chenweiz, rkhardon, lantao}@iu.edu

***Abstract*—Robotic Information Gathering (RIG) relies on the uncertainty of a probabilistic model to identify critical areas for efficient data collection. Gaussian processes (GPs) with stationary kernels have been widely adopted for spatial modeling. However, real-world spatial data typically does not satisfy the assumption of *stationarity*, where different locations are assumed to have the same degree of variability. As a result, the prediction uncertainty does not accurately capture prediction error, limiting the success of RIG algorithms. We propose a novel family of nonstationary kernels, named the Attentive Kernel (AK), which is simple, robust, and can extend any existing kernel to a nonstationary one. We evaluate the new kernel in elevation mapping tasks, where AK provides better accuracy and uncertainty quantification over the commonly used RBF kernel and other popular nonstationary kernels. The improved uncertainty quantification guides the downstream RIG planner to collect more valuable data around the high-error area, further increasing prediction accuracy. A field experiment demonstrates that the proposed method can guide an Autonomous Surface Vehicle (ASV) to prioritize data collection in locations with high spatial variations, enabling the model to characterize the salient environmental features.**[1]

I. Introduction

Collecting informative data for effective modeling has been an active research topic in different domains, including active learning in machine learning [1], optimal experimental design in statistics [2], and optimal placements in sensor networks [3]. *Robotic Information Gathering (RIG)* has recently received increasing attention due to its wide application, including environmental modeling and monitoring [4–13], 3D reconstruction and inspection [14–17], search and rescue [18, 19], autonomous exploration [20–23], and system identification [24–26]. The defining element that distinguishes the aforementioned active information acquisition problems and RIG is the robot embodiment's physical constraint – we cannot "teleport" the robot to an arbitrary sampling location, and data must be collected sequentially along a trajectory. *Informative planning* seeks an action sequence or a policy that yields observations maximizing an information-theoretic objective function under the robot's motion and sensing budget constraints [27–37]. The objective is derived from the uncertainty of probabilistic models such as Gaussian processes (GPs) [38–48], Hilbert maps [49–52], occupancy grid maps [11, 30, 53], and Gaussian mixture models [54–57].

Fig. 1a illustrates the workflow of a RIG system. The three major forces that drive RIG are (a) probabilistic models with well-calibrated uncertainty, (b) information-theoretic objective functions, and (c) informative planners. This work belongs to the first aspect: we aim to improve the uncertainty of GPs, which yields more informative objective functions for RIG. Such fundamental improvements can apply to any informative planner using any objective function.



(a) Ground-Truth Environment

(b) Prediction of RBF (c) Uncertainty of RBF (d) Error of RBF

(e) Prediction of AK (f) Uncertainty of AK (g) Error of AK

Fig. 1. **Comparison of GPR models with RBF kernel and the AK in terrain mapping**. The color indicates elevation, and black dots are training samples. The AK portrays the salient environmental features in more detail and assigns higher uncertainty to the high-error area.

Gaussian process regression (GPR) is one of the leading methods for mapping continuous spatiotemporal phenomena. Stationary kernels, *e.g.*, the RBF kernel and the Matérn family [58], are commonly adopted in a GPR. However, real-world spatial data typically does not satisfy the assumption of *stationarity* – i.e., that different locations have the same degree of variability. For instance, the environment in Fig. 1a has higher variability around the crater. As a result of such a mismatch, GPR with stationary kernels cannot portray the characteristic environmental features in detail. Fig. 1b shows the overs-moothed prediction. The prediction error and uncertainty are inconsistent (*c.f.*, the circled region in Fig. 1c and Fig. 1d), leading to degraded performance if used with RIG.

There is extensive work on GPs with nonstationary data (Section II-B). However, as shown in our experiments, prior work leaves room for improvement. The challenge is that nonstationary models are often too flexible to be trained. To address this, we propose a novel family of nonstationary kernels named the *Attentive Kernel (AK)*. The main ideas

[1] Videos of field/simulated experiments can be found at https://weizhe-chen.github.io/attentive_kernels/

behind the AK are limiting the nonstationary model to *select* among a fixed set of correlation scales and masking out data across sharp transitions by *selecting* subsets of relevant data for each prediction. The "soft" selection process is learned from the data. Fig. 1e shows GPR prediction with the AK on the same dataset. As highlighted by the arrows, the AK depicts the environment at a finer granularity. Fig. 1f and Fig. 1g show that the AK allocates higher uncertainty to the high-error area.

Contributions. The main contribution of this paper is in designing the AK and evaluating its suitability for RIG. We present an extensive evaluation on elevation mapping tasks in several natural environments that exhibit a range of nonstationary features, comparing the AK to the stationary RBF kernel and the leading nonstationary kernels: the Gibbs kernel [59–66] and Deep Kernel Learning (DKL) [67, 68]. The results show a significant advantage of the AK across passive learning, standard active learning, and RIG. We also present a field experiment to demonstrate the behavior of the proposed method in a real-world environment, where the prediction uncertainty of the AK guides an Autonomous Surface Vehicle (ASV) to identify essential sampling locations and collect valuable data rapidly. Last but not least, we release the code for reproducibility[2] and a software library for facilitating future research on RIG[3].

II. Related Work

A. Robotic Information Gathering

Research on RIG mainly revolves around the following three aspects: probabilistic models, information-theoretic objective functions, and informative planning algorithms.

Probabilistic Models and Objectives. Models are primarily discussed in coordinating multiple robots and improving computational efficiency. Jang et al. [44] apply the distributed GPs [69] to decentralized multi-robot online active sensing. Ma et al. [38] and Stachniss et al. [46] use sparse GPs to alleviate the computational burden. Mixture models [70] have been applied to divide the workspace into smaller parts for multiple robots to model an environment simultaneously [42, 43]. The early work by Krause and Guestrin [45] is highly related to our work. They use a spatially varying linear combination of localized stationary processes to model the nonstationary pH values. The weight of each local GP is the normalized predictive variance at the test location. This idea is similar to the lengthscale-selection idea in Section IV-C. The main difference is that they manually partition the workspace while our model learns a weighting function from data. To the best of our knowledge, this paper is the first to discuss the influence on RIG performance brought by the uncertainty quantification capability of probabilistic models. Research on information-theoretic objective functions is dedicated to addressing the computational bottleneck [35, 71, 72].

Informative Planning. Early works on informative planning propose various *recursive greedy* algorithms that provide performance guarantee by exploiting the *submodularity* property of the objective function [73–75]. Planners based on dynamic programming lift the assumption of the objective function at the expense of higher computational complexity [76, 77]. These methods solve combinatorial optimization problems in discrete domains, thus scaling poorly in the problem size. To develop efficient planners in *continuous* space with motion constraints, Hollinger and Sukhatme [27] introduce sampling-based informative motion planning, which is further developed to online variants [31, 34]. Monte Carlo tree search planners are conceptually similar to sampling-based informative planners [78, 79] and have recently garnered great attention [4, 20, 28, 29, 80]. Trajectory optimization is a solid competitor to sampling-based planners. Bayesian optimization [39, 81, 82] and evolutionary strategy [6, 11, 30] are the two dominating methods in this realm. New frameworks of RIG, *e.g.*, imitation learning [32], are constantly emerging.

B. Nonstationary Gaussian Processes and Kernels

GPs suffer from two significant limitations [70]. The first one is the notorious cubic computational complexity of a vanilla implementation. Recent years have witnessed remarkable progress in solving this problem based on sparse GPs [83–85]. The second drawback is that the covariance function is commonly assumed to be stationary, limiting the modeling flexibility. Developing nonstationary GP models that are easy to train is still an active open research problem. Ideas of handling nonstationarity can be roughly grouped into three categories: input-dependent lengthscale [59–65], input warping [67, 68, 86–89], and the mixture of experts [70, 90].

Input-dependent lengthscale provides excellent flexibility to learn different correlation scales at different input locations, but optimizing the lengthscale function is difficult [91]. Input warping is more widely applicable because it endows any stationary kernel with the ability to model nonstationarity by mapping the input locations to a distorted space and assuming stationarity holds in the new space. This approach has a tricky requirement: the mapping must be *injective* to avoid undesirable folding of the space [86, 87, 89]. The Mixture of GP experts (MoGPE) uses a *gating network* to allocate each data to a local GP that learns its hyperparameters from the assigned data. It typically requires Gibbs sampling [70], which can be slow. Hence, one might need to develop a faster approximation [92]. We view MoGPE as an orthogonal direction to other nonstationary GPs/kernels because any GP models can be treated as the experts.

The AK lies at the intersection of these three categories. In Section IV-C, we implement input-dependent lengthscale by weighting base kernels with different prefixed lengthscales at each location. Composing base kernels reduces the difficulty of learning a lengthscale function from scratch and makes our method compatible with any base kernel. In Section IV-D, we augment the input with extra dimensions. We can view the augmentation as warping the input space to a higher-dimensional space, ensuring *injectivity* by design. Combining these two ideas gives a conceptually similar model to

[2] https://github.com/weizhe-chen/attentive_kernels
[3] https://pypolo.readthedocs.io/

MoGPE [70] in that they both divide the space into multiple regions and learn localized hyperparameters. The idea of augmenting the input dimensions has been discussed [93]. However, they treat the augmented vector as a latent variable and resort to Markov chain Monte Carlo (MCMC) for inference. The AK treats the augmentation as a deterministic function of the input, resulting in a more straightforward inference procedure, and can be used in MoGPE to build more flexible models.

In robotic mapping, another line of notable work on probabilistic models is the family of Hilbert maps [49–51], which aims to alleviate the computational bottleneck of the GP occupancy maps [94] by projecting the data to another feature space and applying a logistic regression classifier in the new space. Since Hilbert maps are typically used for occupancy mapping [95] and reconstruction tasks [96], related work also considers nonstationarity for better prediction [52, 97].

III. Problem Statement

Consider deploying a robot to build a map of an *initially unknown* environment *efficiently* using the *sparse* sensing measurements of onboard sensors. For instance, when reconstructing a pollution distribution map, the environmental sensors can only measure the pollutant concentration in a point-wise sampling manner, yielding sparse measurements along the trajectory. Another scenario is to build a sizeable bathymetric map of the seabed. In such a vast space, depth measurements can be viewed as *point measurements* even though the sensor might be a multi-beam sonar. Exhaustively sampling the whole environment is prohibitive, if not impossible; thus, one must develop adaptive planning algorithms to collect the most informative data for building an accurate model. This problem is RIG, *a.k.a.* informative (path/motion) planning, or active/adaptive sensing.

Problem 1. The target environment is an unknown function $\mathtt{f}_{\text{env}}(\mathbf{x}) : \mathbb{R}^D \mapsto \mathbb{R}$ defined over spatial locations $\mathbf{x} \in \mathbb{R}^D$. Let $\mathbb{T} \triangleq \{t\}_{t=0}^{T}$ be the set of decision epochs. A robot at state $\mathbf{s}_{t-1} \in \mathcal{S}$ takes an action $a_{t-1} \in \mathcal{A}$, arrives at the next state $\mathbf{s}_t$ following a transition function $p(\mathbf{s}_t \mid \mathbf{s}_{t-1}, a_{t-1})$, and collects $N_t \in \mathbb{N}$ noisy measurements $\mathbf{y}_t \in \mathbb{R}^{N_t}$ at sampling locations $\mathbf{X}_t = [\mathbf{x}_1, \dots, \mathbf{x}_{N_t}]^\intercal \in \mathbb{R}^{N_t \times D}$. We assume that the transition function is known and deterministic, and the robot state is observable. The robot maintains a probabilistic model built from all the data collected so far $\mathbb{D}_t = \{(\mathbf{X}_i, \mathbf{y}_i)\}_{i=1}^{t}$. The model provides predictive mean $\mu_t : \mathbb{R}^D \mapsto \mathbb{R}$ and predictive variance $\nu_t : \mathbb{R}^D \mapsto \mathbb{R}_{\geq 0}$ functions. Let $\mathbf{x}^\star$ be a test/query location and $\mathtt{error}(\cdot)$ be an error metric. At each decision epoch $t \in \mathbb{T}$, our goal is to find sampling locations that minimize the expected error after updating the model with the collected data

$$\arg\min_{\mathbf{X}_t} \mathtt{E}_{\mathbf{x}^\star} \left[\mathtt{error} \left(\mathtt{f}_{\text{env}}(\mathbf{x}^\star), \mu_t(\mathbf{x}^\star), \nu_t(\mathbf{x}^\star) \right) \right]. \tag{1}$$

Eq. (1) cannot be used as the objective function for a planner because the ground-truth function $\mathtt{f}_{\text{env}}$ is unknown. RIG bypasses this problem by optimizing a surrogate objective.

Problem 2. Assuming the same conditions as Problem 1, find *informative* sampling locations that minimize an information-theoretic objective function $\mathtt{info}(\cdot)$, *e.g.,* entropy:

$$\arg\min_{\mathbf{X}_t} \mathtt{E}_{\mathbf{x}^\star} \left[\mathtt{info} \left(\nu_t(\mathbf{x}^\star) \right) \right]. \tag{2}$$

RIG implicitly assumes that solving Problem 2 can also address Problem 1 well. This assumption is valid when the model uncertainty is *well-calibrated*. A model with well-calibrated uncertainty gives high uncertainty when the prediction error is significant and low uncertainty otherwise. When using GPR with the commonly used stationary kernels to reconstruct a real-world environment, the uncertainty is not well-calibrated because the assumption of stationarity does not hold. Specifically, high uncertainty is assigned to the less sampled areas, regardless of the prediction error (see Figs. 1c, 2 and 11f). Our goal is to develop a kernel to improve the uncertainty quantification and prediction accuracy of GPR.

IV. Methodology

A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution [58]. We place a Gaussian process prior over the function $\mathtt{f}(\mathbf{x}) \sim \mathcal{GP}(\mathtt{m}(\mathbf{x}), \mathtt{k}(\mathbf{x}, \mathbf{x}'))$, which is specified by a mean function $\mathtt{m}(\mathbf{x})$ and a covariance function $\mathtt{k}(\mathbf{x}, \mathbf{x}')$ (a.k.a. kernel). GPR assumes that observations are corrupted by additive Gaussian white noise $p(y|\mathbf{x}) = \mathcal{N}(y|\mathtt{f}(\mathbf{x}), \sigma^2)$, with noise scale σ. This paper focuses on the kernel construction, independent of the inference method in GPs. Therefore, we skip the discussion of inference methods and use the standard maximization of marginal likelihood to optimize the model in our experiments.

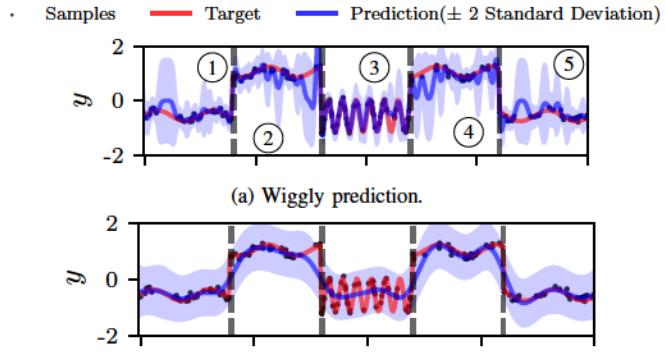
A. Attentive Kernel

We propose the following kernel to deal with nonstationarity. At first glance, this looks like a heuristic composite kernel. However, the following sections will explain how we design this kernel from the first principles. In short, the kernel is distilled from a generative model called AKGPR that models nonstationary processes.

Definition 1 (Attentive Kernel)**.** Given two inputs $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^D$, vector-valued functions $\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x}) : \mathbb{R}^D \mapsto [0,1]^M$ and $\mathbf{z}_{\boldsymbol{\phi}}(\mathbf{x}) : \mathbb{R}^D \mapsto [0,1]^M$ parameterized by $\boldsymbol{\theta}, \boldsymbol{\phi}$, an amplitude α, and a set of M base kernels $\{\mathtt{k}_m(\mathbf{x}, \mathbf{x}')\}_{m=1}^{M}$, let $\bar{\mathbf{w}} = {}^{\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})}/_{\|\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})\|_2}$, and $\bar{\mathbf{z}} = {}^{\mathbf{z}_{\boldsymbol{\phi}}(\mathbf{x})}/_{\|\mathbf{z}_{\boldsymbol{\phi}}(\mathbf{x})\|_2}$. The Attentive Kernel is defined as

$$\mathtt{ak}(\mathbf{x}, \mathbf{x}') = \alpha \bar{\mathbf{z}}^\intercal \bar{\mathbf{z}}' \sum_{m=1}^{M} \bar{w}_m \mathtt{k}_m(\mathbf{x}, \mathbf{x}') \bar{w}'_m. \tag{3}$$

We learn parametric functions that map each input $\mathbf{x}$ to $\mathbf{w}$ and $\mathbf{z}$. $\bar{w}_m \bar{w}'_m$ gives *similarity attention scores* to weight the set of base kernels $\{\mathtt{k}_m(\mathbf{x}, \mathbf{x}')\}_{m=1}^{M}$. The inner product $\bar{\mathbf{z}}^\intercal \bar{\mathbf{z}}'$ defines a *visibility attention score* to mask the kernel value. Definition 1 is generic in that any existing kernel can be the base kernel. To address nonstationarity, we choose the base kernels to be a set of stationary kernels with the same functional form but different lengthscales. Specifically, we use



(a) Wiggly prediction.

(b) Oversmoothed prediction.

Fig. 2. **Learning a nonstationary function using GPR with RBF kernel.** The target function consists of five partitions separated by the dashed lines. The function changes drastically in partition#3 and smoothly in the remaining partitions. The transitions between partitions are sharp. This simple function is challenging for a stationary kernel with a *single* lengthscale hyperparameter. GPR with a stationary RBF kernel produces either the wiggly prediction shown in (a) or the over-smoothed prediction in (b). Note that, in (a), the prediction in the smooth regions is rugged, and the uncertainty is over-conservative when the training sample is sparse. The prediction in (b) only captures the general trend, and every input location seems equally uncertain.

RBF kernels with M evenly spaced lengthscales in the interval $[\ell_{\min}, \ell_{\max}]$:

$$\mathbf{k}_m(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|_2^2}{2\ell_m^2}\right), m = 1, \ldots, M. \quad (4)$$

Note that the lengthscales $\{\ell_m\}_{m=1}^M$ are prefixed constants rather than trainable variables. When applying the attentive kernel to a GPR, we optimize all the hyperparameters $\{\alpha, \theta, \phi, \sigma\}$ by maximizing the marginal likelihood, and make prediction in the standard way.

B. Two Types of Nonstationarity

The example in Fig. 2 motivates us to consider using different lengthscales at different input locations. Ideally, we need a smaller lengthscale for partition#3 and larger lengthscales for the others. In addition, we need to break the correlations among data points in different partitions. An ideal nonstationary model should handle these two types of nonstationarity.

Gibbs [59] and Paciorek and Schervish [60] have shown how one can construct a valid kernel with input-dependent lengthscales, namely, a lengthscale *function*. The standard approach uses another GP to model the lengthscale function, which is then used in the kernel of a GP, yielding a hierarchical Bayesian model. Several papers have developed inference techniques for such models and demonstrated their use in some applications [61–65]. Recently, Remes et al. [66] have shown that modeling the lengthscale function using a neural network improves performance.

However, the parameter optimization of such models is sensitive to data distribution and parameter initialization and leaves room for improvement. To address this, we propose a new approach that *avoids learning a lengthscale function explicitly*. Instead, every input location can (a) *select* among a set of GPs with different predefined primitive lengthscales and (b) *select* which training samples are used when making a prediction. This idea – selecting instead of inferring a localized lengthscale – avoids difficulties in prior work. These ideas are developed in the following sections.

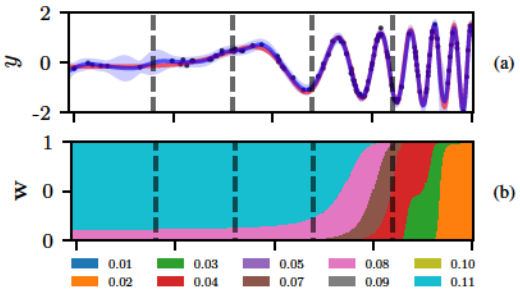


Fig. 3. **Learning $f(x) = x\sin(40x^4)$ with soft lengthscale selection.** The w-plot visualizes each input location's associated weighting vector $\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})$. The more vertical length a color occupies, the higher weight we assign to the GP with the corresponding lengthscale. The learned weighting function gradually shift its weight from smooth GPs to bumpy ones.

C. Lengthscale Selection

Consider a set of M independent GPs with a set of predefined primitive lengthscales $\{\ell_m\}_{m=1}^M$. Intuitively, if every input location can select a GP with an appropriate lengthscale to make prediction, the nonstationarity can be handled well. We can achieve this by an *input-dependent* weighted sum

$$\mathbf{f}(\mathbf{x}) = \sum_m^M \mathbf{w}_m(\mathbf{x})\mathbf{g}_m(\mathbf{x}), \text{ where} \quad (5)$$

$$\mathbf{g}_m(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{k}(\mathbf{x}, \mathbf{x}'|\ell_m)). \quad (6)$$

$\mathbf{w}_m(\mathbf{x})$ is the m-th output of a vector-valued weighting function $\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})$ parameterized by $\boldsymbol{\theta}$.

Consider the extreme case where $\mathbf{w} = [\mathbf{w}_1(\mathbf{x}), \ldots, \mathbf{w}_M(\mathbf{x})]^\intercal$ is an "one-hot" vector – a binary vector with only one element being 1. In this case, $\mathbf{w}$ selects one GP, and hence one lengthscale, depending on the input location. Inference techniques such as Gibbs sampling or Expectation Maximization are often required for learning such discrete "assignment" parameters. We lift this requirement by a continuous relaxation:

$$\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{softmax}(\tilde{\mathbf{w}}_{\boldsymbol{\theta}}(\mathbf{x})). \quad (7)$$

Here, $\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})$ is differentiable *w.r.t.* $\boldsymbol{\theta}$, which can be optimized by the marginal likelihood maximization via gradient ascent. Moreover, using "soft" weights has an advantage in modeling gradually changing nonstationarity, as shown in Fig. 3. Note that dividing the function into several discrete regimes using extreme weights is not reasonable for such a gradually changing function.

Fig. 4 shows that lengthscale selection better predicts the same dataset as in Fig. 2. We can effectively model both the jittery pattern in partition#3 and the gentle variations in the other partitions. However, when facing abrupt changes, as shown in the circled area, the model can only select a very small lengthscale to accommodate the loose correlations among data. If samples near the abrupt change are not dense enough, a small lengthscale might bring us a consequential

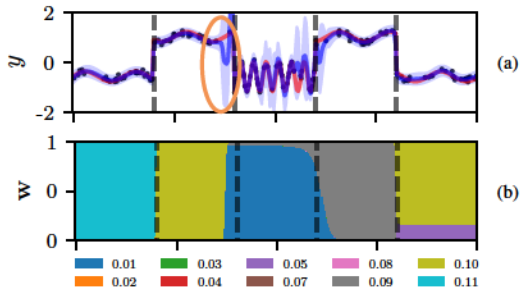


Fig. 4. **Learning the same function as in Fig. 2 using lengthscale selection.**

prediction error. The following section will explain how to handle abrupt changes using instance selection.

D. Instance Selection

Intuitively, an input-dependent lengthscale specifies each data point's neighborhood radius. Changing the radius cannot handle abrupt changes well because data sampled before and after an abrupt change should break their correlations even when they are close. We need to control the *visibility* among samples: each sample learns only from other samples in the same subgroup. To this end, we associate each input with a *membership vector* $\mathbf{z}$ and use the dot product between two membership vectors to control visibility. Two inputs are visible to each other when they hold similar memberships. Otherwise, their correlation will be masked out:

$$\mathrm{k}([\mathbf{x},\mathbf{z}],[\mathbf{x}',\mathbf{z}']) = \mathbf{z}^\intercal\mathbf{z}'\mathrm{k}_{\text{base}}(\mathbf{x},\mathbf{x}'). \tag{8}$$

We can view this as input *dimension augmentation* where we append $\mathbf{z}$ to $\mathbf{x}$ but use a structured kernel in the joint space of $[\mathbf{x},\mathbf{z}]$. It is also helpful to understand the case of extreme-valued hard partitions. In this case, the dot product is equal to 1 if $\mathbf{z}$ and $\mathbf{z}'$ are the same one-hot vector and is 0 otherwise. That is to say, when two points have different memberships, Eq. (8) masks the correlation to zero. In this way, we only use the subset of data points in the same group. To make the model more flexible and simplify the parameter optimization, we use soft memberships:

$$\mathbf{z}_{\boldsymbol{\phi}}(\mathbf{x}) = \mathrm{softmax}(\tilde{\mathbf{z}}_{\boldsymbol{\phi}}(\mathbf{x})). \tag{9}$$

E. The AKGPR Model

Combining the two ideas, we get the AKGPR model. While this is not immediately apparent, we show below that this model can be separated into a standard GPR and the AK (Definition 1). The generative model is as follows.

- Generate $\mathbf{w}$ and $\mathbf{z}$ using Eq. (7) and Eq. (9).
- Compute the kernel values using Eq. (8).
- Generate $g_m \triangleq \mathrm{g}_m(\mathbf{x})$ from the corresponding GP (6).
- Compute $\mathtt{f}(\mathbf{x})$ via Eq. (5).
- Generate y from the Gaussian likelihood.

Parameterization and Optimization. To instantiate an AKGPR model, we must specify the weighting function $\mathbf{w}_{\boldsymbol{\theta}}(\mathbf{x})$ and the membership function $\mathbf{z}_{\boldsymbol{\phi}}(\mathbf{x})$. Our implementation parameterizes these functions using a simple neural network with two hidden layers[4]. We optimize all the parameters by maximizing the log marginal likelihood $\ln p(\mathbf{y}|\sigma,\alpha,\boldsymbol{\theta},\boldsymbol{\phi})$ with a lower learning rate for the neural network parameters.

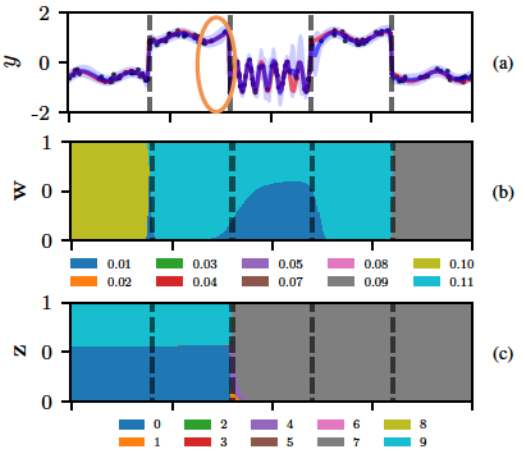


Fig. 5. **Learning the same function as in Fig. 2 using AKGPR. A** weight/membership vector is visualized as a stack of bar plots produced by its elements. Different colors represent different lengthscales/dimensions of the weight/membership vector.

Fig. 5 shows the prediction of the AKGPR on the example from Fig. 2. Now we can model the jittery part, the smooth parts, and the abrupt changes accurately. Compared to Fig. 2 where uncertainty only depends on the density of samples, the uncertainty of AKGPR can better reveal the prediction error. The AKGPR puts more weight on the GPs with small lengthscales in partition#3 and those with large lengthscales in other partitions. Note that the AKGPR switches the membership vector $\mathbf{z}$ in the circled area to mask the inter-partition correlations, which cannot be realized by lengthscale selection in Fig. 4. Due to this modeling advantage, Fig. 5 is qualitatively better than Fig. 4.

F. Analysis

AKGPR is GPR with the AK. By the definition of GPs, the training function values $\mathbf{g}_m \triangleq [\mathrm{g}_m(\mathbf{x}_1),\dots,\mathrm{g}_m(\mathbf{x}_N)]^\intercal$ and the test function value $g_m^\star \triangleq \mathrm{g}_m(\mathbf{x}^\star)$ at arbitrary test input $\mathbf{x}^\star$ jointly follow a multivariate Gaussian distribution. Let $\mathbf{f} \triangleq [\mathtt{f}(\mathbf{x}_1),\dots,\mathtt{f}(\mathbf{x}_N)]^\intercal$ and $f^\star \triangleq \mathtt{f}(\mathbf{x}^\star)$. Aggregate the weights of N training inputs into $\mathbf{w}_m \triangleq [\mathtt{w}_m(\mathbf{x}_1),\dots,\mathtt{w}_m(\mathbf{x}_N)]^\intercal$ and denote $w_m^\star \triangleq \mathtt{w}_m(\mathbf{x}^\star), \mathbf{W}_m = \mathtt{diag}(\mathbf{w}_m)$. Eq. (5) implies that their joint vector is the sum of M linearly transformed multivariate Gaussian variables, which also follows a multivariate Gaussian distribution:

$$\begin{bmatrix}\mathbf{f}\\ f^\star\end{bmatrix} = \sum_m^M \begin{bmatrix}\mathbf{W}_m & \mathbf{0}\\ \mathbf{0}^\intercal & w_m^\star\end{bmatrix}\begin{bmatrix}\mathbf{g}_m\\ g_m^\star\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0},\begin{bmatrix}\mathbf{C} & \mathbf{c}\\ \mathbf{c}^\intercal & c\end{bmatrix}\right), \text{ where}$$

$$\mathbf{C} = \textstyle\sum_{m=1}^M \mathbf{W}_m\mathbf{K}_m\mathbf{W}_m, \tag{10}$$

$$\mathbf{c} = \textstyle\sum_{m=1}^M \mathbf{W}_m\mathbf{k}_m w_m^\star, \tag{11}$$

$$c = \textstyle\sum_m^M w_m^\star k_m w_m^\star. \tag{12}$$

[4]This is an arbitrary choice for the sake of simplicity and modeling flexibility. Any other parametric functions should also work.

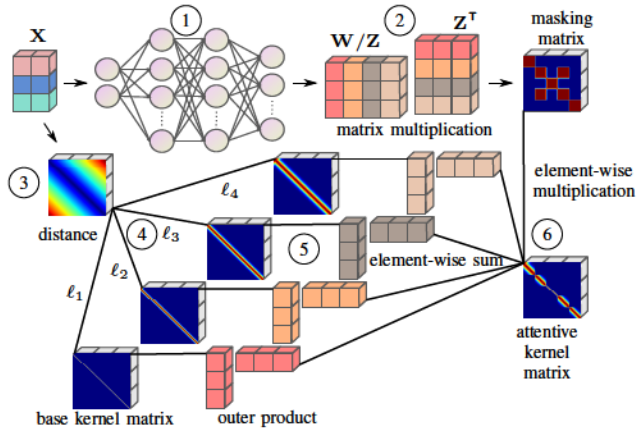


Fig. 6. **Computational diagram of the AK.**

Algorithm 1 RIG with The AK

Arguments: $N_{\max}, \alpha, \sigma, \{\mathtt{k}_m(\mathbf{x}, \mathbf{x}')\}_{m=1}^{M}$
$\mathtt{w}_{\boldsymbol{\theta}}(\mathbf{x}), \mathtt{z}_{\boldsymbol{\phi}}(\mathbf{x}), \mathtt{strategy}$

1: compute normalization and standardization statistics
2: $\mathtt{kernel} \leftarrow \mathtt{AK}(\alpha, \{\mathtt{k}_m(\mathbf{x}, \mathbf{x}')\}_{m=1}^{M}, \mathtt{w}_{\boldsymbol{\theta}}(\mathbf{x}), \mathtt{z}_{\boldsymbol{\phi}}(\mathbf{x}))$
3: $\mathtt{model} \leftarrow \mathtt{GPR}(\mathtt{kernel}, \sigma)$
4: $t \leftarrow 0$
5: **while** $\mathtt{model}.N_{\text{train}} < N_{\max}$ **do** ▷ sampling budget
6: $\mathbf{x}_{\text{info}} \leftarrow \mathtt{strategy}(\mathtt{model})$ ▷ informative waypoint
7: $\mathbf{X}_t, \mathbf{y}_t \leftarrow \mathtt{tracking\_and\_sampling}(\mathbf{x}_{\text{info}})$ ▷ N_t samples
8: $\bar{\mathbf{X}}_t, \bar{\mathbf{y}}_t \leftarrow \mathtt{normalize\_and\_standardize}(\mathbf{X}_t, \mathbf{y}_t)$
9: $\mathtt{model.add\_data}(\bar{\mathbf{X}}_t, \bar{\mathbf{y}}_t)$
10: $\mathtt{model.optimize}(N_t)$ ▷ maximize marginal likelihood
11: $t \leftarrow t + 1$
12: **return** model

The kernel values are given by Eq. (8). Now from Eqs. (10) to (12) we observe that AKGPR is equivalent to a GPR with the AK given in Definition 1.

We normalize $\mathbf{w}$ and $\mathbf{z}$ with ℓ^2-norm to ensure that the maximum kernel value (when $\mathbf{x} = \mathbf{x}'$) is 1, and α is the only parameter that controls the amplitude. Otherwise, the interplay between the amplitude hyperparameter α and the scaling effect of the kernel makes the optimization unstable. The analysis above holds for the normalized versions of $\mathbf{x}, \mathbf{z}$ as well.

Computational Complexity. Kernel matrix computations are typically done in a batch manner to take advantage of the parallelism in linear algebra libraries. Fig. 6 shows the computational diagram of the self-covariance matrix of an input matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$ for the case where $\mathtt{w}_{\boldsymbol{\theta}}(\mathbf{x})$ and $\mathtt{z}_{\boldsymbol{\phi}}(\mathbf{x})$ are parameterized by the same function. The computation of a cross-covariance matrix and the case where $\mathtt{w}_{\boldsymbol{\theta}}(\mathbf{x})$ and $\mathtt{z}_{\boldsymbol{\phi}}(\mathbf{x})$ are parameterized separately are handled similarly. We first pass $\mathbf{X}$ to a two-hidden-layer neural network to get $\mathbf{W} \in \mathbb{R}^{N \times M}$ and $\mathbf{Z} \in \mathbb{R}^{N \times M}$. The computational complexity of this step is $\mathcal{O}(NDH + NH^2 + NHM)$. Then, we compute a visibility masking matrix $\mathbf{O} = \mathbf{Z}\mathbf{Z}^\intercal$, which takes $\mathcal{O}(N^2M)$. After getting the pairwise distance matrix $\big(\mathcal{O}(N^2D)\big)$, we can compute the base kernel matrices using different lengthscales $\big(\mathcal{O}(N^2)\big)$. The m-th kernel matrix is scaled by the outer-product matrix of the m-th column of $\mathbf{W}$, which takes $\mathcal{O}(N^2M)$. Finally, we sum up the scaled kernel matrices and multiply the result with the visibility masking matrix to get the AK matrix $\big(\mathcal{O}(N^2M)\big)$. We defer the discussion of the choices of network size H and number of base kernels M to the sensitivity analysis section Section V-E. In short, these will be relatively small numbers, so the overall computational complexity is still $\mathcal{O}(N^2D)$.

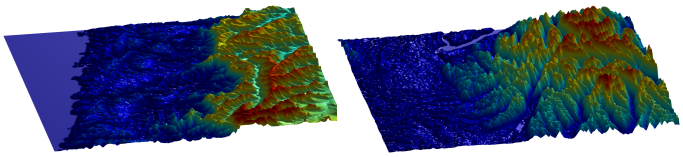
G. RIG with the AK

Algorithm 1 puts AK in the context of RIG. The system requires the following input arguments: the maximum number of training data $N_{\max}$, the initial kernel amplitude α, the initial noise scale σ, a set of M base kernels $\{\mathtt{k}_m(\mathbf{x}, \mathbf{x}')\}_{m=1}^{M}$, functions $\mathtt{w}_{\boldsymbol{\theta}}(\mathbf{x})$, $\mathtt{z}_{\boldsymbol{\phi}}(\mathbf{x})$, and a sampling strategy. First, we need to compute the statistics to normalize the inputs $\mathbf{X}$ to be roughly in the range $[-1, 1]$ and standardize the targets $\mathbf{y}$ to nearly have zero mean and unit variance (line 1). We can get these statistics from prior knowledge of the environment. The workspace extent is typically known, allowing the normalization statistics to be readily calculated. The target-value statistics can be rough estimates or computed from a pilot environment survey [98]. Then, we instantiate an AK and a GPR with the given parameters (lines 2-3). At each decision epoch t, the sampling strategy proposes an informative waypoint based on the predictive entropy of the probabilistic model (line 6). The robot tracks the informative waypoint and collects samples along the trajectory (line 7). The new samples are normalized and standardized and then appended to the model's training set (lines 8-9). Finally, we maximize the log marginal likelihood for N_t iterations (line 10). The robot repeats predicting (hidden in line 6), planning, sampling, and optimizing until the sampling budget is exceeded (line 5).

To reduce the number of parameters and increase training stability, in the experiments, we unify the two functions $\mathtt{w}_{\boldsymbol{\theta}}(\mathbf{x}) \triangleq \mathtt{z}_{\boldsymbol{\phi}}(\mathbf{x})$ and use the same set of parameters $\boldsymbol{\theta} = \boldsymbol{\phi}$, namely, training only one shared network. We discuss the two-network implementation in the ablation study (Section V-E). We also notice that Occam's razor effect in the marginal likelihood is insufficient for preventing overfitting when training nonstationary kernels for many iterations. However, the AK is less prone to overfitting than the Gibbs kernel and DKL (see Appendix E). Tompkins et al. [88] also raised this point in their overfitting analysis. The proper way of training GP models with nonstationary kernels is still an open research problem and has recently received increasing attention [99, 100]. Held-out validation or cross-validation is not suitable for RIG, which does not have access to a large amount of data and has a real-time constraint. We use a rule-of-thumb early-stopping training scheme that works well empirically. Specifically, we train the model on all the collected data $\mathbb{D}_t$ for N_t iterations after collecting N_t samples at the t-th epoch.

V. Experiments

We design our experiments to address the following questions. (Q1) Is the uncertainty quantification of AK better than



(a) N17E073 (b) N47W124

Fig. 7. **Two of the environments used in the elevation mapping tasks**.

its stationary counterpart and the nonstationary baselines? (Q2) Can we achieve better performance in active learning and RIG with the improved uncertainty quantification? (Q3) Is the performance of AK sensitive to the parameter settings? To answer Q1, we use random sampling experiments in Section V-B to ensure that the sampling strategy does not bias the results. For Q2, we conduct both active learning (Section V-C) and RIG experiments (Section V-D) to disentangle the influence of the model and the planner. RIG relies on a planner that considers the physical constraints of the robot embodiment, while active learning is planner-agnostic. Finally, we address Q3 by evaluating the system under different configurations.

A. Experimental Setups

Environments. We have conducted extensive experiments in 4 environments that exhibit various nonstationary features. To shorten the discussion we present two environments here. Complete results are given in Appendix B. Fig. 7 shows the two environments. In Fig. 7a, the environment consists of a flat part, a mountainous area, and a rocky region with many ridges. The right part of Fig. 7b varies drastically, and its left part is relatively flat.

Probabilistic Model. We build a GPR with noisy training samples collected from the ground-truth digital elevation maps in all experiments. The GPR takes two-dimensional sampling locations as inputs and predicts the elevation. We first collect 50 samples to have an initial optimization of the hyperparameters and compute the statistics to normalize the inputs and standardize the targets.

Sampling Strategies. We use different sampling strategies in the three sets of experiments. In random sampling experiments, we draw a sample uniformly at random at each decision epoch. In active sampling experiments, we evaluate the predictive uncertainty on a set of 1000 randomly generated candidate locations and then sample from the location with the highest predictive entropy. While the AK can be plugged into any advanced informative planner for RIG, we use a simple planner to evaluate its performance. Specifically, in addition to the predictive entropy, this planner also computes the distances from these locations to the robot's position and normalizes the predictive entropy and distance to $[0, 1]$, respectively. Each candidate location's informativeness score is defined as the normalized entropy minus the normalized distance. This planner outputs the informative waypoint with the highest score. The robot moves to the waypoint via a tracking controller and samples along the path. Note that the number of collected samples N_t varies at different decision epochs depending on the distance from the robot to the informative waypoint.

Baselines. We compare AK with three popular kernels that have been recently shown to provide good performance. Among the three kernels, two are nonstationary, including the Gibbs kernel and DKL, and the third is the stationary RBF kernel widely used in RIG. Specifically, the Gibbs kernel extends the lengthscale to be any positive function of the input and degenerates to an RBF kernel when using a constant lengthscale function. Following [66] that showed improved results, the lengthscale function is modeled using a neural network instead of using a hierarchical process. DKL addresses nonstationarity through input warping. A neural network transforms the inputs to a feature space where stationary kernels are assumed to be sufficient. We use the same neural network for AK and DKL and change the output dimension to be one for the Gibbs kernel because it requires a scaler-valued lengthscale function.

Metrics. We care about the prediction performance and whether the predictive uncertainty can effectively reflect the prediction error. Following standard practice in the GP literature, we use *standardized mean squared error (SMSE)* and *mean standardized log loss (MSLL)* to measure these quantities. SMSE is the mean squared error divided by the variance of test targets. With this standardization the trivial method of guessing the mean of the training targets has an SMSE of approximately 1. To take the predictive uncertainty into account, one can evaluate the negative log probability, i.e., log loss, of a test target,

$$-\ln p(y^\star|\mathbb{D}, \mathbf{x}^\star) = \frac{\ln(2\pi\nu)}{2} + \frac{(y^\star - \mu)^2}{(2\nu)},$$

where μ and ν are the mean and variance in the predictive distribution. MSLL standardizes the log loss by subtracting the loss obtained under the trivial model, which predicts using a Gaussian with the mean and variance of the training targets. The MSLL will be approximately zero for simple methods and negative for better methods. In the experiments, we also measured the root-mean-square error (RMSE), mean negative log-likelihood (MNLL), and mean absolute error (MAE). The results are consistent across different metrics, and the complete results with all metrics are given in Appendix B. We report the mean and standard deviation of the metrics over 10 runs of the experiments with different random seeds.

B. Random Sampling

Figs. 8a and 8b show that AK has better a prediction error than the other kernels with randomly sampled data. In these experiments, we are especially interested in the MSLL because a lower MSLL means that the model gives high uncertainty when its prediction is far from the test target, which can help active sampling and informative planning reduce the error faster. Figs. 8c and 8d show that AK has a significant advantage in MSLL over the other methods. The Gibbs kernel also has some advantage over the other two methods, but the MSLL of DKL is almost the same as that of RBF.

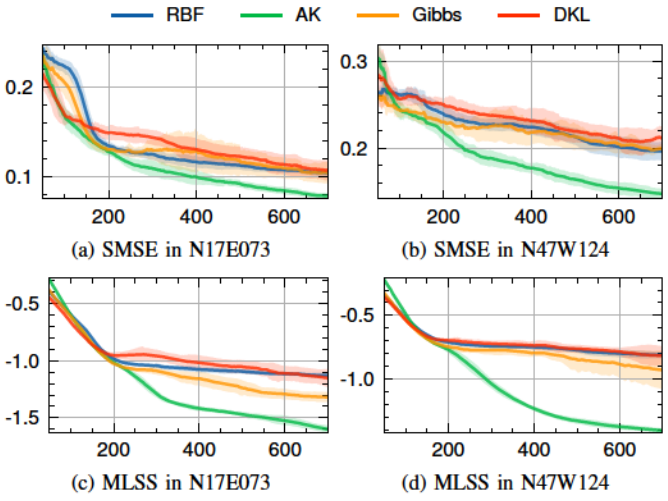


Fig. 8. **SMSE and MSLL vs the number of samples in random sampling.**

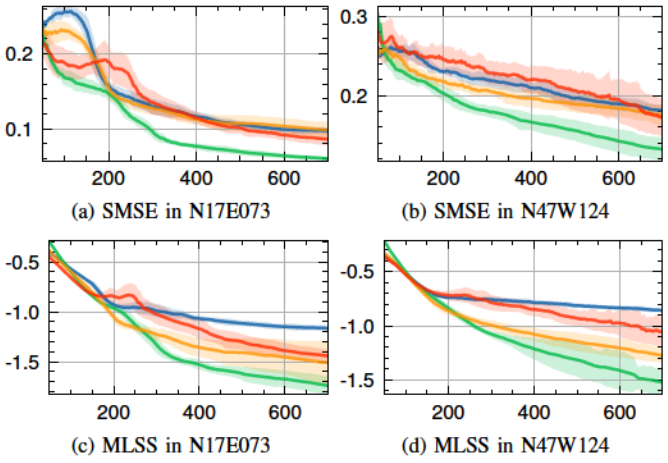


Fig. 9. **SMSE and MSLL vs the number of samples in active sampling.**

C. Active Sampling

Figs. 9a and 9b show that AK also has faster error reduction when the samples are actively collected. The AK can quickly identify the crucial areas that account for most of the error and sample more valuable data in those spots, leading to a significant gap in the final metrics. In Figs. 9c and 9d, Gibbs and DKL improve over RBF in uncertainty quantification in the active sampling.

D. Informative Planning

Informative planning is a more challenging task than active learning because once the robot decides to visit an informative waypoint, it has to collect the samples along its trajectory. Given a fixed maximum number of samples, the number of decision epochs of RIG is much smaller than that of active sampling, which makes informed decisions more essential. Fig. 10 shows that AK is consistently leading in all the metrics with the informative planning strategy. The Gibbs kernel still has clear improvement over the RBF kernel in MSLL, but DKL falls short in these experiments.

Fig. 11 shows a snapshot of the prediction results of

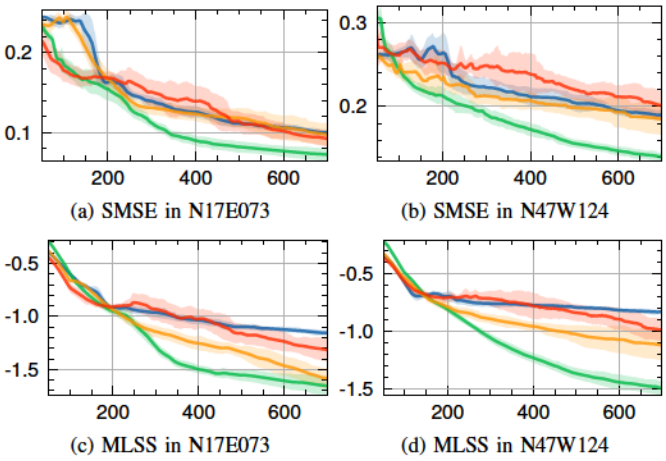


Fig. 10. SMSE and MSLL of robotic information gathering experiments.

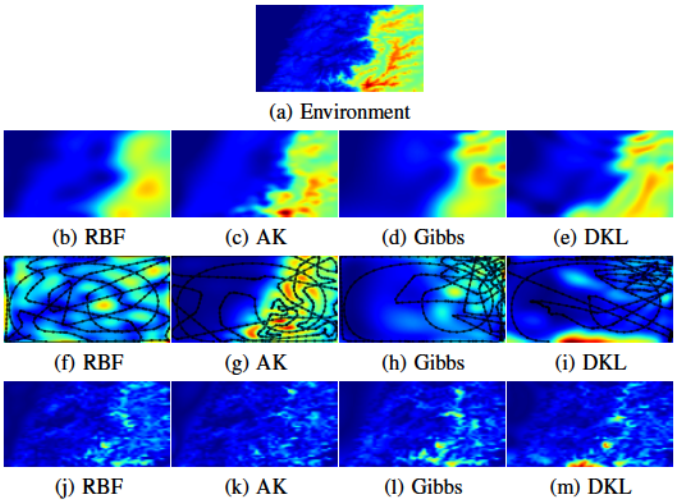


Fig. 11. **Snapshots of RIG with different kernels.** (b-e) show the prediction maps, (f-i) are the uncertainty maps with sampling paths, and (j-m) present the absolute error maps.

different methods after 400 samples, along with the ground-truth environment in the first row. The RBF kernel misses many environmental features that nonstationary kernels can capture. We observe the following behaviors by comparing the patterns in the uncertainty maps and error maps. Note that the error maps use the same color scale for ease of comparison across different methods. Each uncertainty map has its color scale – red color only indicates relatively high uncertainty within the map.

- Regardless of the prediction errors, the RBF kernel gives the less sampled area higher uncertainty. The sampling path uniformly covers the space.
- The AK assigns higher uncertainty in the regions with more significant error. The sampling path focuses more on the complex region.
- The Gibbs kernel also has higher uncertainty in the rocky region but does not assign high uncertainty to the lower-right. Therefore, the sampling path concentrates on the upper-right corner and misses some high-error spots.

(a) System (b) Workspace (c) Final Snapshot

Fig. 12. **An active elevation mapping field experiment**.

- When using DKL, the robot also samples the upper-right corner densely. The prediction error at the bottom of the map is the largest across different methods. However, DKL also places high uncertainty there.

E. Sensitivity Analysis and Ablation Study

We stress-test the AK under different parameter settings for sensitivity analysis and compare four variants of the AK for ablation study. We present the conclusions of the analysis here and provide full details of these tests in Appendices C and D.

Sensitivity Analysis. Increasing the number of base kernels or primitive lengthscales M improves performance, albeit with a diminishing return and higher computational cost. Choosing M in the range of $[5, 10]$ is a good tradeoff between performance and computational efficiency. AK is not sensitive to the number of hidden units H in the neural network as long as H is not too small, e.g., only two units. Using smaller minimum lengthscales ℓ_{min} yields better performance, but the advantage of choosing a ℓ_{min} smaller than 0.01 is negligible, so 0.01 is an appropriate choice. AK is also robust to the setting of maximum lengthscale ℓ_{max} as long as it is not too small. After normalizing the input to be nearly in the range $[-1, 1]$, choosing ℓ_{max} between $[0.5, 1.0]$ is suitable. Overall, the AK is robust to various parameter settings.

Ablation Study. The ablation study shows that lengthscale selection is necessary. Dropping it decreases the performance significantly. On the other hand, we do not observe a significant performance advantage from instance selection using the current training scheme. Nonetheless, as illustrated in Fig. 5, we expect instance selection to provide better modeling of sharp transitions. Since instance selection improves the prediction only in a small region, the improvement might not be evident in the aggregated evaluation metrics. Using two separate neural networks does not provide an improvement. However, it deteriorates the uncertainty quantification in one environment. We conjecture that the two-network implementation might show its strength with a more refined approach to parameter training.

F. Field Experiment

We demonstrate the proposed AK in a RIG task – active elevation/bathymetric mapping. We deploy an ASV with a single-beam sonar pointing downward to collect depth measurements (Fig. 12a). The robot can localize itself by fusing the GPS and IMU data and actuate through the two thrusters. Our goal is to build an elevation map within the workspace shown in Fig. 12b with a small number of samples. From the satellite imagery, we can vaguely see interesting environmental features in the lower-left and upper-right corners of the workspace. Fig. 12c shows a snapshot of the final model prediction, uncertainty, and sampling path. The prediction uncertainty is effectively reduced after sampling, and most of the samples are collected in critical regions with drastic elevation variations. Such a biased sampling pattern allows us to model the general trend of smooth regions with a small number of samples while capturing the characteristic environmental features at a fine granularity.

VI. Conclusion and Future Work

In this paper, we investigate the uncertainty quantification of probabilistic models, which is decisive for the performance of RIG but has received little attention. We present a family of nonstationary kernels called the Attentive Kernel, which is simple, robust, and can extend any stationary kernel to a nonstationary one. An extensive evaluation of elevation mapping tasks shows that AK provides better accuracy and uncertainty quantification than baselines. The improved uncertainty quantification guides the informative planning algorithms to collect more valuable samples around the high-error area, thus further reducing the prediction error. A field experiment demonstrates that AK enables an ASV to collect more samples in important sampling locations and capture the salient environmental features. The results indicate that misspecified probabilistic models affect the RIG performance profoundly. Future work includes further investigating the influence of *outliers* and *heteroscedastic noise* on RIG. Besides, a more principled training scheme of nonstationary kernels can be an essential future research direction.

VII. Acknowledgement

We acknowledge the support of NSF with grant numbers 1906694, 2006886, and 2047169. We are also grateful for the computational resources provided by the Amazon AWS Machine Learning Research Award. The constructive comments by the anonymous reviewers are greatly appreciated. We thank Durgakant Pushp and Mahmoud Ali for their help when conducting the field experiment.

References

[1] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[2] Anthony C. Atkinson. The usefulness of optimum experimental designs. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):59–76, 1996.

[3] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. *The Journal of Machine Learning Research (JMLR)*, 9:235–284, June 2008.

[4] 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.

[5] 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.

[6] 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.

[7] Kai-Chieh Ma, Lantao Liu, Hordur K. Heidarsson, and Gaurav S. Sukhatme. Data-driven learning and planning for environmental sampling. *Journal of Field Robotics (JFR)*, 35(5):643–661, 2018.

[8] 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.

[9] 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.

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

[11] Marija Popović, Gregory Hitz, Juan Nieto, Inkyu Sa, Roland Siegwart, and Enric Galceran. Online informative path planning for active classification using UAVs. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5753–5758, 2017.

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

[13] Shi Bai, Tixiao Shan, Fanfei Chen, Lantao Liu, and Brendan Englot. Information-Driven Path Planning. *Current Robotics Reports*, pages 1–12, 2021.

[14] 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.

[15] Nikolay Atanasov, Bharath Sankaran, Jerome Le Ny, Geoge J. Pappas, and Kostas Daniilidis. Nonmyopic view planning for active object classification and pose estimation. *IEEE Transactions on Robotics (T-RO)*, 30 (5):1078–1090, 2014.

[16] 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.

[17] 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)*, 2021.

[18] 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.

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

[20] 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.

[21] Tung Dang, Christos Papachristos, and Kostas Alexis. Visual saliency-aware receding horizon autonomous exploration with application to aerial robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2526–2533, 2018.

[22] Sebastian Thrun, Scott Thayer, William Whittaker, Christopher Baker, Wolfram Burgard, David Ferguson, Dirk Hahnel, D Montemerlo, Aaron Morris, Zachary Omohundro, et al. Autonomous exploration and mapping of abandoned mines. *IEEE Robotics & Automation Magazine (RAM)*, 11(4):79–91, 2004.

[23] Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *Conference on Robot Learning (CoRL)*, 2021.

[24] Alexandre Capone, Gerrit Noske, Jonas Umlauft, Thomas Beckers, Armin Lederer, and Sandra Hirche. Localized active learning of Gaussian process state space models. In *Conference on Learning for Dynamics and Control (L4DC)*, volume 120, pages 490–499,

2020.

[25] Mona Buisson-Fenet, Friedrich Solowjow, and Sebastian Trimpe. Actively learning Gaussian process dynamics. In *Learning for Dynamics and Control (L4DC)*, pages 5–15, 2020.

[26] Marc Deisenroth and Carl E Rasmussen. PILCO: a model-based and data-efficient approach to policy search. In *International Conference on Machine Learning (ICML)*, pages 465–472, 2011.

[27] Geoffrey A Hollinger and Gaurav S Sukhatme. Sampling-based robotic information gathering algorithms. *The International Journal of Robotics Research (IJRR)*, 33(9):1271–1287, 2014.

[28] 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.

[29] Weizhe Chen and Lantao Liu. Pareto Monte Carlo tree search for multi-objective informative planning. In *Robotics: Science and Systems (RSS)*, 2019.

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

[31] 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.

[32] Sanjiban Choudhury, Mohak Bhardwaj, Sankalp Arora, Ashish Kapoor, Gireeja Ranade, Sebastian Scherer, and Debadeepta Dey. Data-driven planning via imitation learning. *The International Journal of Robotics Research (IJRR)*, 37(13-14):1632–1672, 2018.

[33] Ryan A MacDonald and Stephen L Smith. Active sensing for motion planning in uncertain environments via mutual information policies. *The International Journal of Robotics Research (IJRR)*, 38(2-3):146–161, 2019.

[34] 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.

[35] 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.

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

[37] Haruki Nishimura and Mac Schwager. SACBP: belief space planning for continuous-time dynamical systems via stochastic sequential action control. *The International Journal of Robotics Research (IJRR)*, 40(10-11): 1167–1195, 2021.

[38] 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.

[39] 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.

[40] Roman Marchant and Fabio Ramos. Bayesian optimisation for informative continuous path planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6136–6143, 2014.

[41] Maani Ghaffari Jadidi, Jaime Valls Miro, and Gamini Dissanayake. Gaussian processes autonomous mapping and exploration for range-sensing mobile robots. *Autonomous Robots (AURO)*, 42(2):273–290, 2018.

[42] 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.

[43] 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.

[44] 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.

[45] Andreas Krause and Carlos Guestrin. Nonmyopic active learning of Gaussian processes: an exploration-exploitation approach. In *International Conference on Machine learning (ICML)*, pages 449–456, 2007.

[46] Cyrill Stachniss, Christian Plagemann, and Achim J Lilienthal. Learning gas distribution models using sparse Gaussian process mixtures. *Autonomous Robots*, 26(2):187–202, 2009.

[47] Marija Popović, Teresa Vidal-Calleja, Jen Jen Chung, Juan Nieto, and Roland Siegwart. Informative path planning for active field mapping under localization uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 10751–10757, 2020.

[48] Jongseok Lee, Jianxiang Feng, Matthias Humt, Marcus Gerhard Müller, and Rudolph Triebel. Trust your robots! predictive uncertainty estimation of neural networks with sparse gaussian processes. In *Conference on Robot Learning (CoRL)*, pages 1168–1179, 2022.

[49] Fabio Ramos and Lionel Ott. Hilbert maps: scalable

continuous occupancy mapping with stochastic gradient descent. *The International Journal of Robotics Research (IJRR)*, 35(14):1717–1730, 2016.

[50] Ransalu Senanayake and Fabio Ramos. Bayesian Hilbert maps for dynamic continuous occupancy mapping. In *Conference on Robot Learning (CoRL)*, pages 458–471, 2017.

[51] Vitor Guizilini and Fabio Ramos. Variational Hilbert regression for terrain modeling and trajectory optimization. *The International Journal of Robotics Research (IJRR)*, 38(12-13):1375–1387, 2019.

[52] Ransalu Senanayake, Anthony Tompkins, and Fabio Ramos. Automorphing kernels for nonstationarity in mapping unstructured environments. In *Conference on Robot Learning (CoRL)*, pages 443–455, 2018.

[53] Manish Saroya, Graeme Best, and Geoffrey A Hollinger. Roadmap learning for probabilistic occupancy maps with topology-informed growing neural gas. *IEEE Robotics and Automation Letters (RA-L)*, 6(3):4805–4812, 2021.

[54] Wennie Tabib, Kshitij Goel, John Yao, Mosam Dabhi, Curtis Boirum, and Nathan Michael. Real-time information-theoretic exploration with gaussian mixture model maps. In *Robotics: Science and Systems (RSS)*, 2019.

[55] Aditya Dhawale and Nathan Michael. Efficient parametric multi-fidelity surface mapping. In *Robotics: Science and Systems (RSS)*, volume 2, page 5, 2020.

[56] Micah Corah, Cormac O'Meadhra, Kshitij Goel, and Nathan Michael. Communication-efficient planning and mapping for multi-robot exploration in large environments. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):1715–1721, 2019.

[57] Cormac O'Meadhra, Wennie Tabib, and Nathan Michael. Variable resolution occupancy mapping using gaussian mixture models. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):2015–2022, 2018.

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

[59] Mark N Gibbs. *Bayesian Gaussian processes for regression and classification*. PhD thesis, University of Cambridge, 1997.

[60] Christopher J. Paciorek and Mark J. Schervish. Nonstationary covariance functions for Gaussian process regression. In *Advances on Neural Information Processing Systems (NeurIPS)*, 2003.

[61] Sami Remes, Markus Heinonen, and Samuel Kaski. Non-stationary spectral kernels. In *Advances on Neural Information Processing Systems (NeurIPS)*, 2017.

[62] Markus Heinonen, Henrik Mannerström, Juho Rousu, Samuel Kaski, and Harri Lähdesmäki. Non-stationary Gaussian process regression with Hamiltonian Monte Carlo. In *The Journal of Machine Learning Research (JMLR)*, volume 51, pages 732–740, 2016.

[63] Tobias Lang, Christian Plagemann, and Wolfram Burgard. Adaptive non-stationary kernel regression for terrain modeling. In *Robotics: Science and Systems (RSS)*, 2007.

[64] Christian Plagemann, Sebastian Mischke, Sam Prentice, Kristian Kersting, Nicholas Roy, and Wolfram Burgard. Learning predictive terrain models for legged robot locomotion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3545–3552, 2008.

[65] Christian Plagemann, Kristian Kersting, and Wolfram Burgard. Nonstationary Gaussian process regression using point estimates of local smoothness. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-KDD)*, pages 204–219. Springer, 2008.

[66] Sami Remes, Markus Heinonen, and Samuel Kaski. Neural non-stationary spectral kernel. *arXiv*, 2018.

[67] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep Kernel Learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 370–378, 2016.

[68] Roberto Calandra, Jan Peters, Carl Edward Rasmussen, and Marc Peter Deisenroth. Manifold Gaussian processes for regression. In *International Joint Conference on Neural Networks (IJCNN)*, pages 3338–3345, 2016.

[69] Marc Deisenroth and Jun Wei Ng. Distributed Gaussian processes. In *International Conference on Machine Learning (ICML)*, pages 1481–1490, 2015.

[70] Carl Rasmussen and Zoubin Ghahramani. Infinite mixtures of Gaussian process experts. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 14, 2001.

[71] Benjamin Charrow, Sikang Liu, Vijay Kumar, and Nathan Michael. Information-theoretic mapping using Cauchy-Schwarz quadratic mutual information. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4791–4798, 2015.

[72] 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 *Robotics: Science and Systems (RSS)*, volume 11, pages 3–12, 2015.

[73] 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.

[74] 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.

[75] 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.

[76] 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, 2009.

[77] 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.

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

[79] 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.

[80] 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.

[81] 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.

[82] 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.

[83] 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.

[84] 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.

[85] 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.

[86] Paul D. Sampson and Peter Guttorp. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association (JASA)*, 87(417):108–119, 1992.

[87] Jasper Snoek, Kevin Swersky, Rich Zemel, and Ryan Adams. Input warping for Bayesian optimization of non-stationary functions. In *International Conference on Machine Learning (ICML)*, volume 32, pages 1674–1682, 2014.

[88] Anthony Tompkins, Rafael Oliveira, and Fabio T Ramos. Sparse Spectrum Warped Input Measures for Nonstationary Kernel Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 16153–16164, 2020.

[89] Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep Gaussian processes. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[90] Martin Trapp, Robert Peharz, Franz Pernkopf, and Carl Edward Rasmussen. Deep structured mixtures of gaussian processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 2251–2261, 2020.

[91] Kangrui Wang, Oliver Hamelijnck, Theodoros Damoulas, and Mark Steel. Non-separable non-stationary random fields. In *International Conference on Machine Learning (ICML)*, pages 9887–9897, 2020.

[92] Duy Nguyen-Tuong, Jan Peters, and Matthias Seeger. Local Gaussian process regression for real time online model learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 21, 2008.

[93] Tobias Pfingsten, Malte Kuss, and Carl Edward Rasmussen. Nonstationary Gaussian process regression using a latent extension of the input space. In *International Society for Bayesian Analysis (ISBA)*, 2006.

[94] Simon T O'Callaghan and Fabio T Ramos. Gaussian process occupancy maps. *The International Journal of Robotics Research (IJRR)*, 31(1):42–62, 2012.

[95] Kevin Doherty, Jinkun Wang, and Brendan Englot. Probabilistic map fusion for fast, incremental occupancy mapping with 3D Hilbert maps. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1011–1018, 2016.

[96] Vitor Guizilini and Fabio Ramos. Learning to reconstruct 3D structures for occupancy mapping. In *Robotics: Science and Systems (RSS)*, 2017.

[97] Anthony Tompkins, Ransalu Senanayake, and Fabio Ramos. Online domain adaptation for occupancy mapping. In *Robotics: Science and Systems (RSS)*, 2020.

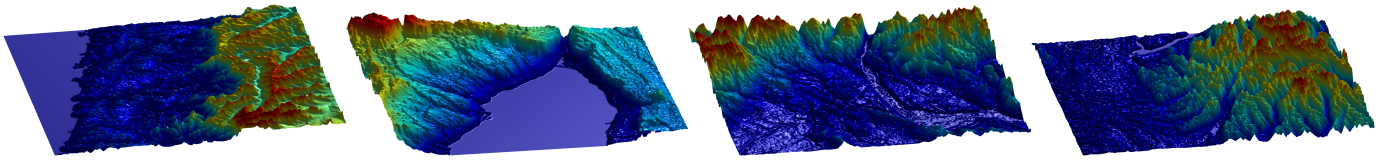
[98] Stephanie Kemna, Oliver Kroemer, and Gaurav S Sukhatme. Pilot surveys for adaptive informative sampling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6417–6424, 2018.

[99] Sebastian W Ober, Carl E Rasmussen, and Mark van der Wilk. The promises and pitfalls of deep kernel learning. In *Uncertainty in Artificial Intelligence (UAI)*, pages 1206–1216, 2021.

[100] Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*, 2021.

APPENDIX

A. Environments



(a) N17E073 (b) N43W080 (c) N45W123 (d) N47W124

Fig. 13. **The four environments used in the elevation mapping tasks**. Red means high elevation, and blue represents low elevation.

Fig. 13 shows all the environments used in the experiments. N17E073 consists of a flat part, a mountainous area, and a rocky region with many ridges. N43W080 presents a sharp elevation change at the north part while the lakebed is virtually flat. In N45W123, the environment has a complex upper part and a smoother lower part. There is also a river passing by from the middle. The right part of N47W124 varies drastically, and its left part is relatively flat.

B. Benchmarking Tables

TABLE I
RANDOM SAMPLING PERFORMANCE.

Environment	Method	SMSE$\downarrow_0^1$	MSLL$\downarrow^0$	NLPD$\downarrow$	RMSE$\downarrow_0$	MAE$\downarrow_0$
N17E073	RBF	(1.33±0.03)×10^{-1}	(–9.9 ±0.1) ×10^{-1}	4.59±0.01	(2.33±0.03)×10^{1}	(1.69±0.03)×10^{1}
	AK	**(1.11±0.04)×10^{-1}**	**–1.24 ±0.01**	**4.34±0.01**	**(2.13±0.04)×10^{1}**	**(1.50±0.02)×10^{1}**
	Gibbs	(1.33±0.01)×10^{-1}	–1.09 ±0.02	4.50±0.03	(2.33±0.09)×10^{1}	(1.66±0.04)×10^{1}
	DKL	(1.37±0.06)×10^{-1}	(–9.7 ±0.3) ×10^{-1}	4.62±0.03	(2.37±0.05)×10^{1}	(1.68±0.04)×10^{1}
N43W080	RBF	(7.1 ±0.3) ×10^{-2}	–1.43 ±0.02	3.87±0.02	(1.23±0.03)×10^{1}	8.13 ±0.06
	AK	**(6.0 ±0.5) ×10^{-2}**	**–1.69 ±0.06**	**3.62±0.06**	**(1.11±0.05)×10^{1}**	**7.0 ±0.2**
	Gibbs	(7.2 ±0.4) ×10^{-2}	–1.48 ±0.06	3.83±0.06	(1.25±0.05)×10^{1}	8.3 ±0.3
	DKL	(6.6 ±0.8) ×10^{-2}	–1.49 ±0.04	3.81±0.04	(1.19±0.07)×10^{1}	7.5 ±0.3
N45W123	RBF	(1.65±0.07)×10^{-1}	(–9.4 ±0.3) ×10^{-1}	4.37±0.03	(1.97±0.04)×10^{1}	(1.28±0.03)×10^{1}
	AK	**(1.41±0.06)×10^{-1}**	**–1.28 ±0.02**	**4.03±0.02**	**(1.80±0.04)×10^{1}**	**(1.15±0.02)×10^{1}**
	Gibbs	(1.8 ±0.1) ×10^{-1}	–1.08 ±0.01	4.24±0.02	(2.07±0.07)×10^{1}	(1.34±0.02)×10^{1}
	DKL	(2.0 ±0.1) ×10^{-1}	(–9.1 ±0.1) ×10^{-1}	4.41±0.01	(2.18±0.07)×10^{1}	(1.42±0.06)×10^{1}
N47W123	RBF	(2.26±0.07)×10^{-1}	(–7.2 ±0.1) ×10^{-1}	4.77±0.01	(2.77±0.04)×10^{1}	(1.97±0.02)×10^{1}
	AK	**(1.90±0.05)×10^{-1}**	**–1.06 ±0.01**	**4.43±0.01**	**(2.53±0.03)×10^{1}**	**(1.77±0.02)×10^{1}**
	Gibbs	(2.21±0.08)×10^{-1}	(–7.7 ±0.4) ×10^{-1}	4.72±0.05	(2.74±0.05)×10^{1}	(1.94±0.03)×10^{1}
	DKL	(2.34±0.08)×10^{-1}	(–7.1 ±0.2) ×10^{-1}	4.78±0.02	(2.82±0.05)×10^{1}	(1.98±0.03)×10^{1}

TABLE II
ACTIVE SAMPLING PERFORMANCE.

Environment	Method	SMSE$\downarrow_0^1$	MSLL$\downarrow^0$	NLPD$\downarrow$	RMSE$\downarrow_0$	MAE$\downarrow_0$
N17E073	RBF	(1.41±0.04)×10^{-1}	(–9.8 ±0.2) ×10^{-1}	4.61±0.02	(2.38±0.03)×10^{1}	(1.70±0.03)×10^{1}
	AK	**(1.01±0.02)×10^{-1}**	**–1.32 ±0.04**	**4.36±0.02**	**(2.00±0.02)×10^{1}**	**(1.43±0.02)×10^{1}**
	Gibbs	(1.37±0.06)×10^{-1}	–1.20 ±0.08	4.59±0.03	(2.35±0.06)×10^{1}	(1.72±0.05)×10^{1}
	DKL	(1.33±0.07)×10^{-1}	–1.09 ±0.05	4.59±0.03	(2.32±0.06)×10^{1}	(1.62±0.05)×10^{1}
N43W080	RBF	(7.8 ±0.2) ×10^{-2}	–1.41 ±0.01	3.96±0.01	(1.28±0.01)×10^{1}	9.0 ±0.1
	AK	**(5.1 ±0.2) ×10^{-2}**	**–1.72 ±0.02**	**3.74±0.03**	**(1.02±0.02)×10^{1}**	**6.9 ±0.1**
	Gibbs	(8.0 ±0.6) ×10^{-2}	–1.48 ±0.05	3.98±0.06	(1.31±0.06)×10^{1}	9.8 ±0.4
	DKL	(7 ±1) ×10^{-2}	–1.6 ±0.1	3.9 ±0.1	(1.2 ±0.1) ×10^{1}	8.2 ±0.6
N45W123	RBF	(1.47±0.04)×10^{-1}	(–9.7 ±0.1) ×10^{-1}	4.36±0.01	(1.85±0.02)×10^{1}	(1.23±0.02)×10^{1}
	AK	**(1.08±0.03)×10^{-1}**	**–1.55 ±0.04**	**4.16±0.02**	**(1.57±0.03)×10^{1}**	**(1.14±0.03)×10^{1}**
	Gibbs	(1.29±0.06)×10^{-1}	–1.48 ±0.05	4.30±0.02	(1.73±0.04)×10^{1}	(1.28±0.02)×10^{1}
	DKL	(1.6 ±0.1) ×10^{-1}	–1.18 ±0.04	4.35±0.03	(1.91±0.07)×10^{1}	(1.35±0.04)×10^{1}
N47W124	RBF	(2.15±0.05)×10^{-1}	(–7.5 ±0.1) ×10^{-1}	4.75±0.01	(2.70±0.03)×10^{1}	(1.90±0.03)×10^{1}
	AK	**(1.78±0.08)×10^{-1}**	**–1.09 ±0.07**	**4.56±0.01**	**(2.45±0.06)×10^{1}**	**(1.75±0.03)×10^{1}**
	Gibbs	(2.04±0.06)×10^{-1}	(–9.9 ±0.5) ×10^{-1}	4.71±0.02	(2.63±0.04)×10^{1}	(1.86±0.03)×10^{1}
	DKL	(2.2 ±0.1) ×10^{-1}	(–8.1 ±0.5) ×10^{-1}	4.76±0.05	(2.75±0.09)×10^{1}	(1.94±0.05)×10^{1}

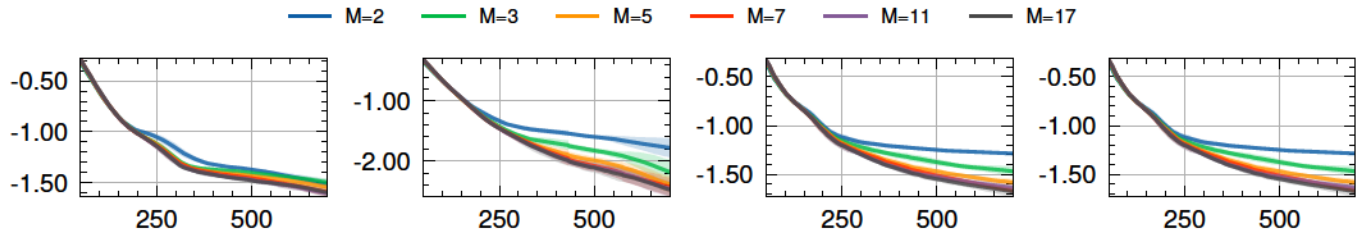
To have a more evident quantitative comparison, we present all the benchmarking results in Tables I to III. Each number summarizes the metric curves by averaging the curves over the x-axis (*i.e.*, the number of samples). This number indicates the

TABLE III
ROBOTIC INFORMATION GATHERING PERFORMANCE.

Environment	Method	SMSE$\downarrow_0^1$	MSLL$\downarrow^0$	NLPD$\downarrow$	RMSE$\downarrow_0$	MAE$\downarrow_0$
N17E073	RBF	(1.45±0.03)×10^{-1}	(−9.7 ±0.2) ×10^{-1}	4.63±0.02	(2.42±0.02)×10^{1}	(1.73±0.02)×10^{1}
	AK	**(1.14±0.04)×10^{-1}**	**−1.27 ±0.03**	**4.41±0.04**	**(2.14±0.04)×10^{1}**	**(1.51±0.02)×10^{1}**
	Gibbs	(1.43±0.07)×10^{-1}	−1.16 ±0.04	4.61±0.04	(2.40±0.07)×10^{1}	(1.76±0.06)×10^{1}
	DKL	(1.38±0.09)×10^{-1}	−1.01 ±0.06	4.61±0.04	(2.38±0.08)×10^{1}	(1.67±0.06)×10^{1}
N43W080	RBF	(7.7 ±0.4) ×10^{-2}	−1.40 ±0.02	3.94±0.02	(1.27±0.03)×10^{1}	8.8 ±0.2
	AK	**(6.6 ±0.2) ×10^{-2}**	**−1.64 ±0.04**	**3.78±0.03**	**(1.14±0.02)×10^{1}**	**7.69 ±0.09**
	Gibbs	(7.6 ±0.9) ×10^{-2}	−1.50 ±0.05	3.91±0.07	(1.25±0.07)×10^{1}	9.0 ±0.6
	DKL	(7.0 ±0.1) ×10^{-2}	−1.56 ±0.07	3.85±0.06	(1.19±0.08)×10^{1}	8.1 ±0.6
N45W123	RBF	(1.60±0.06)×10^{-1}	(−9.3 ±0.2) ×10^{-1}	4.39±0.02	(1.93±0.04)×10^{1}	(1.29±0.02)×10^{1}
	AK	**(1.32±0.06)×10^{-1}**	**−1.43 ±0.04**	**4.15±0.03**	**(1.71±0.04)×10^{1}**	**(1.21±0.03)×10^{1}**
	Gibbs	(1.38±0.07)×10^{-1}	−1.34 ±0.04	4.30±0.03	(1.79±0.05)×10^{1}	(1.32±0.04)×10^{1}
	DKL	(1.7 ±0.2) ×10^{-1}	−1.06 ±0.08	4.41±0.06	(1.99±0.09)×10^{1}	(1.40±0.06)×10^{1}
N47W124	RBF	(2.23±0.06)×10^{-1}	(−7.4 ±0.1) ×10^{-1}	4.76±0.01	(2.75±0.03)×10^{1}	(1.94±0.02)×10^{1}
	AK	**(1.85±0.04)×10^{-1}**	**−1.10 ±0.03**	**4.48±0.03**	**(2.50±0.03)×10^{1}**	**(1.79±0.03)×10^{1}**
	Gibbs	(2.12±0.08)×10^{-1}	(−9.0 ±0.5) ×10^{-1}	4.73±0.03	(2.69±0.05)×10^{1}	(1.91±0.02)×10^{1}
	DKL	(2.36±0.06)×10^{-1}	(−7.7 ±0.4) ×10^{-1}	4.78±0.03	(2.83±0.03)×10^{1}	(1.99±0.04)×10^{1}

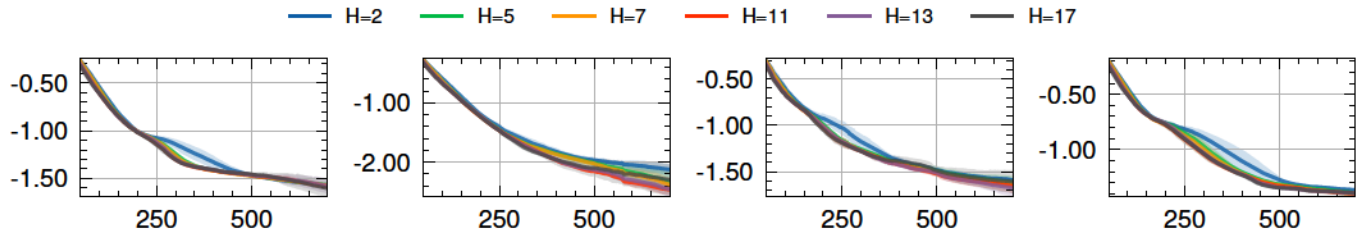
averaged *area under the curve*. A smaller area implies a faster drop in the curve. We can clearly see that AK has significantly better prediction accuracy (*i.e.*, SMSE, MSE, and MAE) and uncertainty quantification (*c.f.*, MSLL and NLPD).

C. Sensitivity Analysis



(a) MSLL in N17E073 (b) MSLL in N43W080 (c) MSLL in N45W123 (d) MSLL in N47W124

Fig. 14. **Sensitivity analysis of the choice of M.**



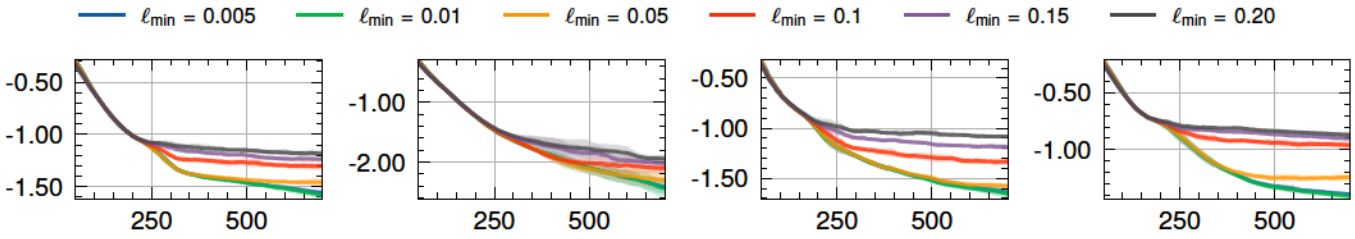
(a) MSLL in N17E073 (b) MSLL in N43W080 (c) MSLL in N45W123 (d) MSLL in N47W124

Fig. 15. **Sensitivity analysis of the choice of H.**

Fig. 14 presents the results of sensitivity analysis of the number of base kernels M, which should be larger than 2. Increasing M brings better performance, albeit with a diminishing return and higher computational complexity. Choosing a number in $\{5, 10\}$ is a good tradeoff between performance and computational efficiency. Fig. 15 shows that the AK is not sensitive to the number of hidden units in the neural network as long as H is not too small. In Fig. 16, smaller minimum lengthscales yield better performance with a diminishing return. The blue line and the green line are overlapped, which means that the advantage is negligible when choosing a minimum lengthscale smaller than 0.01. Therefore, 0.01 is an appropriate choice. As shown in Fig. 17, the AK is robust to the choice of the maximum lengthscale as long as it is not too small. If the inputs are normalized to $[-1, 1]$,choosing a value in the range $[0.5, 1.0]$ is reasonable.

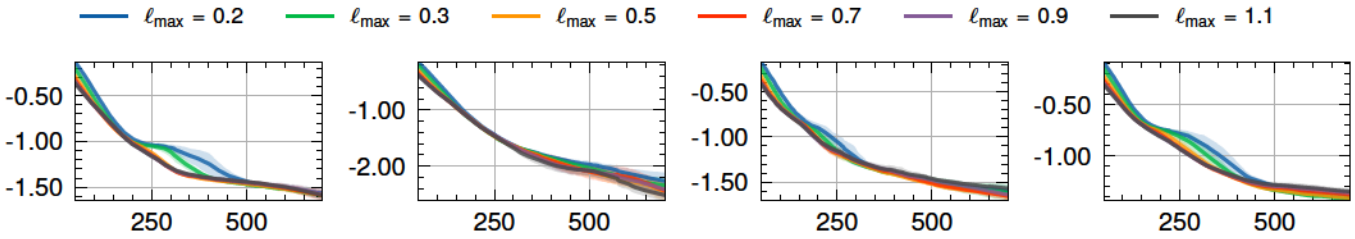
D. Ablation Study

We compare four variants of the attentive kernel in the random sampling experiments for the ablation study. *Full* means the AK presented in the paper, *Weight* represents the AK with only lengthscale selection, *Mask* stands for instance selection



(a) MSLL in N17E073 (b) MSLL in N43W080 (c) MSLL in N45W123 (d) MSLL in N47W124

Fig. 16. **Sensitivity analysis of the choice of ℓ_{min}.**



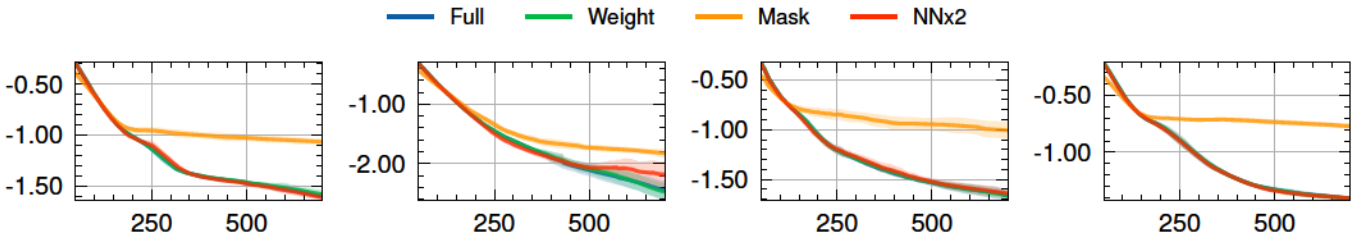
(a) MSLL in N17E073 (b) MSLL in N43W080 (c) MSLL in N45W123 (d) MSLL in N47W124

Fig. 17. **Sensitivity analysis of the choice of ℓ_{max}.**

alone, and *NNx2* uses two separated neural networks. The results show that lengthscale selection is necessary, and dropping it decreases the performance significantly (see the *Mask* line). We do not observe a significant performance advantage from instance selection. Using two separate neural networks for the weighting function and the membership function does not provide an improvement but deteriorates the uncertainty quantification in one environment (*i.e.*, N43W080).

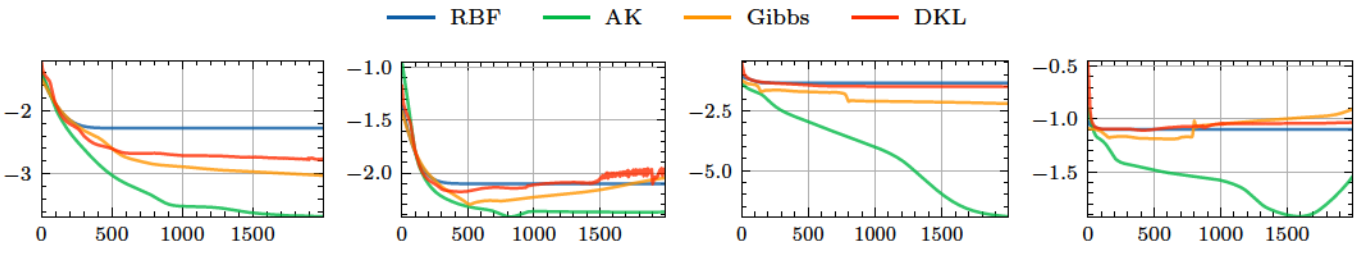
E. Overfitting Analysis

Fig. 19 shows the training and test MSLL. We have repeated the analysis in other environments, but only two representative environments are presented here for compactness. In some environments, as shown in Figs. 19a and 19b, the AK is fairly robust while the Gibbs kernel and DKL show a clear overfitting trend. However, all the nonstationary kernels suffer from overfitting after training for many iterations. The cause of the difference seems to be related to sharp changes in the environment.



(a) MSLL in N17E073 (b) MSLL in N43W080 (c) MSLL in N45W123 (d) MSLL in N47W124

Fig. 18. **Results of the four variants in the ablation study.**



(a) Training MSLL in Volcano (b) Test MSLL in Volcano (c) Training MSLL in N17E073 (d) Test MSLL in N17E073

Fig. 19. **Results of the overfitting analysis** in the *Volcano* environment introduced in Fig. 1a and N17E073.