Efficiently Identifying Hotspots in a Spatially Varying Field with Multiple Robots

Varun Suryan¹ and Pratap Tokekar²

Abstract—In this paper, we present algorithms to identify environmental hotspots using mobile sensors. We examine two approaches: one involving a single robot and another using multiple robots coordinated through a decentralized robot system. We introduce an adaptive algorithm that does not require precise knowledge of Gaussian Processes (GPs) hyperparameters, making the modeling process more flexible. The robots operate for a pre-defined time in the environment. The multi-robot system uses Voronoi partitioning to divide tasks and a Monte Carlo Tree Search for optimal path planning. Our tests on synthetic and a real-world dataset of Chlorophyll density from a Pacific Ocean sub-region suggest that accurate estimation of GP hyperparameters may not be essential for hotspot detection, potentially simplifying environmental monitoring tasks.

I. INTRODUCTION

Mobile robots are increasingly used in collecting information in multitudes of scenarios. For example, a farmer can send a robot to collect the measurements of organic matter in different sub-regions of the farm [1], [2] to understand the soil chemistry [3]. Robots can detect any environmental anomalies, such as a chemical spill in a water body which can have a significant impact on marine life [4]. An aerial robot (Figure 1) can be used to monitor relatively large areas at once [5]. In another application, robots can be deployed in a nuclear power plant to monitor potential leakages by measuring radiation levels [6]. By identifying the sites of higher nuclear radiation using robot sensors, we can efficiently find any potential leakage. In these scenarios, one would be better off just by identifying the hotspot (i.e., maxima) instead of learning the entire environment accurately like in our prior work [2].



Fig. 1. An unmanned aerial vehicle (UAV) flying over a lake to find the chemical spill hotspots [5].

Our goal is to plan the paths to identify the hotspots with a single as well as multiple mobile robots. For a single robot,

we present a Monte Carlo Tree Search (MCTS)-based [7] planning algorithm that uses an Upper Confidence Bound (UCB)-style [8] exploration and works with or without the knowledge of true Gaussian Processes (GP) hyperparameters. In general, GP hyperparameters are optimized during the process and can be a computationally prohibitive task. For the multi-robot case, we present a dynamic partitioning scheme that splits the environment amongst the robots such that no robot is required to cover an especially large portion of the environment. However, instead of partitioning the environment just based on the size, we use the GP estimates and the size of the environment to determine the partitions. Specifically, our partitioning is based on Voronoi tessellation [9] and the UCB metric [8], [10]. This partitioning scheme can work with several planners and find hotspots efficiently. We also allow for the robots to operate in a decentralized fashion with periodic connectivity for coordination.

II. RELATED WORK

The hotspot identification issue aligns with problems like source-seeking in Informative Path Planning literature [11], [12]. Chen and Liu introduced Pareto MCTS, an anytime multi-objective planning method addressing exploration vs. exploitation [13]. While many informative planning studies assume known hyperparameters [2], [14]-[17], online planning estimates them during execution. Binney et al. [16] used initial run data for estimation. Kemna et al. [18] utilized pilot surveys for hyperparameter initialization, accounting for their time in overall planning. MCTS has been commonly used in informative path planning and hotspot identification [7], [13], [19]. They have been shown to have consistencies in balancing the exploration-exploitation efficiently in many applications [20], [21]. Our algorithm AdaptGP-MCTS uses GP-UCB values as the reward heuristics and balances the exploration-exploitation. The performance of UCB planners has been shown to be sensitive with respect to β value [22]. In this work, we use a squared root growth of β which has been proved to achieve better performance on terminal regret [23].

Multi-Robot Systems (MRS) have been actively deployed in precision agriculture [24], [25], and environmental monitoring and exploration [26]–[28]. One of the major challenges in MRS is dividing the task between robots efficiently, especially in practical scenarios when the robots operate in a decentralized manner [29]. Voronoi partitioning is a common approach for multi-robot coordination used in various domains, such as exploration and mapping with ground vehicles, including spatial partitioning [30]–[34]. Kemna et

¹Varun Suryan. Department of Computer Science, University of Maryland College Park, USA. varun.suryan.01@gmail.com

²Pratap Tokekar. Department of Computer Science, University of Maryland, Collge Park, USA. tokekar@umd.edu

al. used a dynamic Voronoi partitioning approach based on the entropy in a decentralized fashion [35]. They repeatedly calculate weighted Voronoi partitions for the space. Each vehicle then runs informative adaptive sampling within its partition. The vehicles can share information periodically. Wenhao et al. presented an adaptive sampling algorithm for learning the density function in multi-robot sensor coverage problems using a Mixture of GP models [36].

III. PROBLEM FORMULATION

We assume that the spatial field under consideration defined over a 2-dimensional environment $U \in \mathbb{R}^2$ is an instance of a GP, F. F is defined by a covariance function of the form,

$$C_Z(x,x') = \sigma^2 \exp\left(-\frac{(x-x')^2}{2l^2}\right); \forall x,x' \in U, \tag{1}$$

defined by a squared-exponential kernel and the hyperparameters σ^2 and l are not known.

Problem 1 (Terminal Regret): Given an operating time budget T, plan a trajectory under budget T for a mobile robot that obtains measurements from U, and reports the location of maxima of the spatial field f at the end, *i.e.*,

minimize
$$f(x^*) - f(\hat{x})$$
,
subject to $len(\tau) + n\eta \le T$.

 τ denotes the tour of the robot. The robot travels at unit speed, obtains one measurement in η units of time, and collects n total measurements. \hat{x} is the location of the maxima of the predicted field while x^* is the location of the maxima of the true spatial field. We do not know x^* and we also do not know f. We only know the GP prediction \hat{f} . The task is to use \hat{f} to be able to predict x^* .

Problem 2 (Multi-robot Hotspot ID): Given an operating time budget T, plan a set of trajectories under budget T for a set of k mobile robots that obtain measurements from the environment U, and report the location of maxima of the spatial field f at the end, *i.e.*,

$$\begin{aligned} & \text{minimize} & & f(x^*) - f(\hat{x}), \\ & \text{subject to} & & \max_{i \in \{1, \dots, k\}} len(\tau_i) + n_i \eta \leq T. \end{aligned}$$

 τ_i denotes the tour of the i^{th} robot. Robots travel with unit speed and obtain one measurement in η units of time. Here, let i^{th} robot collect n_i total measurements.

IV. ALGORITHMS

We start with the algorithm for a single robot followed by the multi-robot version.

A. Single Robot

AdaptGP-MCTS (Algorithm 1) shows the main function that calls the planner MCTS shown in Line 4. Once the planner gives the next measurement location, the robot goes there and collects the measurement. AdaptGP-MCTS monotonically decreases the length scale and monotonically increases the signal variance so that the GP model can capture more complex function candidates [37]. Eliminating

the need to optimize hyperparameters at each step by using AdaptGP-MCTS alleviates the cubic complexity of the hyperparameter optimization. AdaptGP-MCTS starts with an initial σ_0 and l_0 of the GP hyperparameters. The new updated values of hyperparameters are used to get the mean and variance estimate in the next iteration in Line 3. In Line 5, we collect the measurement at location x_t . This measurement is perturbed by the sensor noise ε modeled as a standard normal distribution with mean zero mean and ω^2 variance. ω^2 is assumed to be known *a priori*. Once the operating budget is exhausted, we do a full GP hyperparameter optimization (Line 8). Finally, the location of the predicted maxima is reported (Line 10) where the posterior mean attains its maximum value.

Algorithm 1 AdaptGP-MCTS

```
1: Input: Initial hyperparameters \sigma_0 = 1 and \mathbf{l_0} = diam(Env), \mathbf{X} = \{\}, \mathbf{y} = \{\}, Planner().

2: while t \leq \text{Total time budget } T

3: \hat{\mu}_t(x), \hat{\sigma}_t(x) \leftarrow GP.Predict(\mathbf{X}, \mathbf{y})

4: x_t \leftarrow Planner(\hat{\mu}_t(x), \hat{\sigma}_t(x), t)

5: y_t = f(x_t) + \varepsilon

6: \mathbf{X}.append(x_t); \mathbf{y}.append(y_t)

7: Update \sigma_t = \sigma_0 \log(t); \mathbf{l_t} = \mathbf{l_0}/\log(t)

8: Do a full GP hyperparameter optimization with (\mathbf{X}, \mathbf{y})

9: Estimate the posterior mean \hat{\mu}

10: return argmax_{x \in U} \hat{\mu}(x)
```

Now we discuss the planner which is based on the idea of MCTS and uses GP-UCB values as the reward heuristics. The pseudocode for the planner is given in the Algorithm 2. In the Backpropagation step, we use the GP-UCB values to

```
Algorithm 2 GP-MCTS
```

```
1: Input: \hat{\mu}_{t}(x), \hat{\sigma}_{t}(x), t.

2: while within budget

3: v' \leftarrow \text{Selection}(v)

4: v_{new} \leftarrow \text{Expansion}(v')

5: r_{\mu} + \beta^{1/2} r_{\sigma} \leftarrow \text{Simulation}(v_{new})

6: Backpropagation (v_{new}, r_{\mu} + \beta^{1/2} r_{\sigma})

7: end procedure

8: function Selection(v)

9: while v is fully expanded

10: v \leftarrow \operatorname{argmax}_{child \in v.children} \frac{Q(child)}{n_{child}} + 2\sqrt{\frac{\log(n_{v})}{n_{child}}}

11: return v
```

update the values for ancestral nodes. For reward calculation, we use a root squared growth of $\beta^{1/2}$ in terms of the number of measurements collected:

- 1) Mean: To encourage the exploitation, *i.e.*, $r_{\mu} = \hat{\mu}_t(x)$,
- 2) Variance: To encourage the exploration, *i.e.*, $r_{\sigma} = \hat{\sigma}_t(x)$.

B. Multiple Robots

Our multi-robot algorithm uses Voronoi regions for dynamic partitioning after each epoch.

Definition 1: Given a set of points $p_1, p_2, ..., p_n$ in the plane S, a Voronoi diagram divides the plane S into n Voronoi regions with the following properties [9]:

- Each point p_i lies in exactly one region.
- If a point $q \in S$ lies in the same region as p_i , then the Euclidean distance from p_i to q will be shorter than the Euclidean distance from p_j to q, where p_j is any other point in S.

The points p_1, \ldots, p_n are called generator points for the Voronoi partitions. We use UCB values defined in [8] (the denominator in Equation 2) as the weights from our GP model to estimate the weighted centroids of a Voronoi cell. Let $(x_1^1, x_2^1), \ldots, (x_1^m, x_2^m)_i$ be the set of m points in i^{th} Voronoi partition. Then its centroid can be calculated as follows,

 $Centroid(Vor_i) =$

$$\sum_{k=1}^{k=m} \frac{(x_1^k, x_2^k)_i (\hat{\mu}_t(x_1^k, x_2^k)_i + \beta_t \hat{\sigma}_t(x_1^k, x_2^k)_i)}{\hat{\mu}_t(x_1^k, x_2^k)_i + \beta_t \hat{\sigma}_t(x_1^k, x_2^k)_i}.$$
 (2)

Here, $\hat{\mu}_t(x_1^k, x_2^k)_i$, and $\hat{\sigma}_t(x_1^k, x_2^k)_i$ are the GP mean and variance at location $(x_1^k, x_2^k)_i$ respectively, and β_t is the parameter that controls the exploration-exploitation.

In Algorithm 3, robots operate for n epochs and take m steps per epoch. They begin from set start points. Voronoi regions for these robots are derived using their current positions. During an epoch, each robot's path is mapped out using the Planner() function within its specific Voronoi area. Within an epoch, robots cannot exchange information. Thus, the Planner() function relies solely on the data each robot individually knows during that epoch. Measurements taken are identified by robot number; e.g., $(x_1^t, x_2^t)_1, (x_1^t, x_2^t)_2$ represent data collected by Robots 1 and 2 at time t respectively. Once the epoch concludes, robots share data, and we update the collective GP model, GPcombined, with all cumulatively gathered measurements. Voronoi partitions are then recalculated with current robot positions (Line 5). If the GP hyperparameters are not known, the AdaptGP-MCTS planner for a single robot, detailed in Algorithm 1, can be applied.

V. EMPIRICAL EVALUATION

We start by presenting our empirical results with the case where the GP hyperparameters are assumed to be known. We call this strategy TrueGP-MCTS.

Our tests use Chlorophyll density data from a Pacific Ocean square sub-region, covering longitude from -155.5 to -129.5 and latitude from 9.0 to 35. We modeled an environment using these coordinates, studying a synthetic spatial field. Locations within are treated as search tree nodes. Robots at any location have five motion primitives, uniformly distributed in the $\left[-\frac{\pi}{4}, \frac{\pi}{4}\right]$ range, acting as current node children. The MCTS build iteration cap is 50. We used a random policy for roll-outs, back-propagating average GP-UCB values as rewards. Roll-outs don't have fixed simulation steps. Instead, they're based on the remaining time budget minus the node's depth from the root. This

approach promotes more exploration early on but diminishes as the mission progresses and the environment becomes familiar [7].

An instance of an MCTS tree for a robot is shown in Figure 2. The green arrows represent the entire tree and the blue arrows represent the best trajectory based on this built tree. The blue path shows the robot path until that moment in time and the background heatmap represents the learned GP mean by the robot of the underlying spatial field. For the Expansion Step in Algorithm 2 (Line 4), we expand randomly on any of the unvisited children.

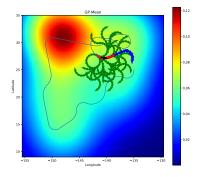


Fig. 2. The robot has five motion primitives.

A. Synthetic Field

We construct a complex spatial field (Figure 3) that has four locations of maxima, three of which are local maxima. For our experiments, we start the robot near the lower left corner from (-149.0, 16.0) so as to trick it into collecting measurements and spending time near one of the local maxima. The actual hotspot is located near the top right corner at (-135.6, 29) where the field attains a maximum value of 1 and a minimum value of 0. We estimated the

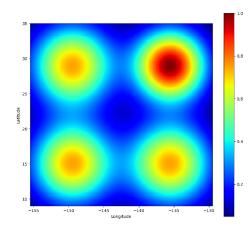


Fig. 3. The environment has four locations of maxima, three of which are local maxima.

hyperparameters *apriori* using a 30×30 grid on this field and minimizing the negative log marginal likelihood of the

Algorithm 3 Voronoi-TrueGP-MCTS

```
1: procedure
 2: Input: Hyperparameters \sigma_0 and l_0, Planner().
   for epoch = [1:n]
           Create k individual copies GP1, GP2, \dots, GP_k of the combined GP model GP_{combined}
 4:
           Calculate Voronoi regions using robot locations as the point generators
 5:
 6:
           for t = [1:m]:
 7:
                  Plan and execute the next robot step computed using Planner() in their respective Voronoi region.
 8:
                  Collect the next environment samples, (x_1^t, x_2^t)_1, \dots, (x_1^t, x_2^t)_k
                  Update the GP models: GP1.update(x_1^t, x_2^t)_1, GP2.update(x_1^t, x_2^t)_2 \dots, GPk.update(x_1^t, y_2^t)_k
 9:
           Combine the samples from all robots and update the combined GP model using all the samples.
10:
11: end procedure
```

TABLE I

The time budget is 350 units with the first subcolumn displaying the BST pattern and the second showing TrueGP-MCTS.

	BST	TrueGP-MCTS
Terminal Regret	11.7130 ± 4.8586	5.3964 ± 2.1146
Avg Cumulative Regret	63.6402 ± 0.7974	54.8814 ± 1.9327
RMSE	11.9767 ± 4.2813	8.2699 ± 1.0573
Distance	19.7206 ± 9.3801	9.7927 ± 4.8182

values at those grid locations. The GP squared-exponential hyperparameters σ_0 , l1, l2, ω^2 for this field were estimated to be 0.251, 5.04, 5.04, 10^{-5} respectively.

The sensor noise standard deviation was set to 0.05 (5% of the spatial field range). The robot plans the path using an MCTS planner with GP-UCB values as the node rewards where the GP variance was multiplied with the square root of $2\sqrt{t}\log\left(\frac{|D|\pi^2}{6\delta}\right)$ as β_t (termed as GP-UCBE in the plots). Here, |D| denotes the number of grid locations used for estimating the GP mean and variance. We used a grid of resolution 130×130 . Hence, |D| is 16900 in our case and we choose δ to be equal to 0.1 [8]. We run ten missions for the robot that starts from (-149.0, 16.0). We compare the performance of the TrueGP-MCTS planner with a Boustrophedon (BST) path.

Table I shows the average mission Percent Terminal Regret, Percent Average Cumulative Regret, Percent Root Mean Squared Error (RMSE) all with respect to the range of the spatial field (*i.e.*, 1), Percent Distance with respect to the diagonal of the environment. The TrueGP-MCTS outperforms the BST on all metrics. The BST exhibits a higher standard deviation in its performance, influenced by the orientation of its pattern, which might occasionally lead to quick hotspot detection or prolonged searches. In contrast, TrueGP-MCTS maintains a more consistent, uniform exploration of the environment.

Figure 4 shows the same metrics as Table I. We can see that in the beginning, BST and TrueGP-MCTS have almost the same performance in terms of terminal regret and distance. However, with a medium budget, the TrueGP-MCTS explores the environment efficiently and converges quickly to report the hotspot location.

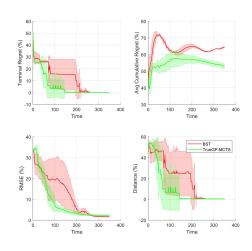
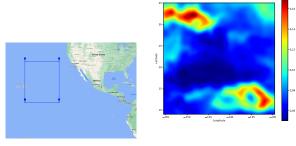


Fig. 4. The sensor noise standard deviation was set to 5%.

B. Chlorophyll Dataset

We evaluate the performance of our algorithms on a real-world dataset of Chlorophyll concentration measured on Oct 8, 2021, obtained from NASA Earth Observations from a Pacific Ocean subregion shown in Figure 5(a). The actual Chlorophyll concentration (mg/m^3) is shown in Figure 5(b). The data collected is from a square region spanning the geographical coordinates, longitude expansion from -155.5 to -129.5, and latitude expansion from 9 to 35 (Figure 5(a)) at 0.5 degree geo-coordinate grid resolution. To query a value at any non-grid location, we used a radial basis function for interpolating and assumed that the interpolated values were the true values at that non-grid location.

The hotspot is located at (-148.67, 32.11) where the Chlorophyll density attains the maximum value equal to 0.17 mg/m^3 and the lowest density value is $0.05 \ mg/m^3$. We estimated the hyperparameters *apriori* using a 30×30 grid on this field and minimizing the negative log marginal likelihood of the values at those grid locations. The GP squared-exponential hyperparameters σ_0 , l1, l2, ω^2 for this field were estimated to be 0.0483, 2.33, 1.99, 10^{-5} respectively. The sensor values are simulated as a normal distribution with the mean as the actual value at the measurement location. The sensor noise standard deviation was set to 0.006 (5% of



(a) Geographical Subregion (b) Chlorophyll Concentration (mg/m^3)

Fig. 5. The environment has longitude expansion from -155.5 to -129.5 and latitude expansion from 9 to 35.

TABLE II
The time budget is 350 units.

	BST	TrueGP-MCTS
Terminal Regret	32.21 ± 16.38	26.37 ± 7.76
Avg Cumulative Regret	76.52 ± 1.01	68.1 ± 1.91
RMSE	26.59 ± 5.16	20.58 ± 1.08
Distance	34.54 ± 12.14	16.84 ± 8.57

the spatial field range).

We run ten missions for the robot that starts from (-142, 18). This starting location was chosen closer to the local maxima and is more likely to trick the robot from identifying the actual hotspot. We compare the performance of the TrueGP-MCTS planner with a Boustrophedon (BST) path. Table II shows all the metrics similar to Table I for the Chlorophyll dataset. The TrueGP-MCTS planner outperforms Boustrophedon on Terminal Regret, RMSE, and Distance and comparably on Cumulative Regret. The TrueGP-MCTS outperforms the BST path and keeps accumulating cumulative regret by continuously exploring the environment even though it has already found the hotspot. Hence, while it might not be always traveling in the highvalue regions (resulting in a higher cumulative regret) its GP-Mean estimate still has the maxima aligned with the actual hotspot location.

Figure 6 shows the same metrics as Table I. In the beginning, BST and TrueGP-MCTS have almost the same performance in terms of terminal regret and distance. However, with a medium budget, the TrueGP-MCTS explores the environment efficiently and converges quickly to report the hotspot location.

C. Unknown GP Hyperparameters

We now present the AdaptGP-MCTS and compare its performance with TrueGP-MCST (known hyperparameters) and OptGP-MCTS (optimized at every timestep). We did the experiments with a single robot on the synthetic spatial field. Table III displays metrics akin to Table I. Over ten missions, TrueGP-MCTS initially performs better than the rest, with a marginal lead over OptGP-MCTS. OptGP-MCTS shows notable initial variability, likely due to its hyperparameters being path-dependent, causing variations across missions.

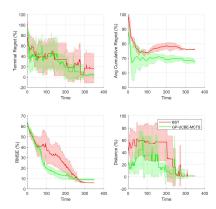


Fig. 6. The sensor noise standard deviation was set to 5%.

TABLE III

The top, middle, and bottom sub-tables display metrics for the time budgets 1-100, 101-200, and 201-350, respectively.

58.02 ± 13.10	17.58 ± 8.30	17.15 ± 3.31
59.27 ± 4.62	53.50 ± 2.80	50.70 ± 2.46
34.33 ± 4.67	22.19 ± 4.98	21.06 ± 3.73
48.38 ± 4.13	27.64 ± 16.46	28.88 ± 5.21
18.13 ± 14.51	2.33 ± 3.34	0.56 ± 0.66
62.58 ± 0.96	59.71 ± 3.12	54.25 ± 3.85
8.92 ± 1.00	5.98 ± 1.09	4.43 ± 0.47
21.23 ± 20.68	4.38 ± 7.20	1.10 ± 1.26
3.13 ± 2.07	0.07 ± 0.10	0.17 ± 0.22
62.39 ± 1.14	57.19 ± 1.61	53.88 ± 1.11
6.36 ± 0.29	2.62 ± 0.38	2.71 ± 0.24
3.28 ± 2.84	0.30 ± 0.21	0.44 ± 0.34

Hence, on a low operating time budget and unknown hyperparameters, one can use OptGP-MCTS. Figure 7 shows the cumulative GP operations time versus the operating budget. TrueGP-MCTS and AdaptGP-MCTS have almost the same computation time but OptGP-MCTS complexity increases significantly. However, as the robot spends more time in the environment, the AdaptGP-MCTS catches up and the performance difference diminishes to less than 3%.

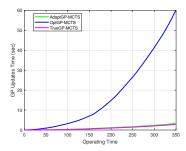


Fig. 7. The sensor noise standard deviation was set to 5%.

D. Multiple Robots

We conducted tests using four robots on the synthetic spatial field depicted in Figure 3. The robots start near the bottom left, enticing them to gather measurements near one of the peaks. We compared three algorithms in each scenario:

TABLE IV

Averaged over 15 trials for four robots and the noise was set to 5%.

	No Partition	Voronoi
1 Hotspot	3.70 ± 1.15	4.60 ± 1.77
2 Hotspots	25.40 ± 8.09	11.40 ± 5.22
3 Hotspots	37.40 ± 6.96	21.10 ± 5.70
2 Hotspots	55.70 ± 14.10	34.20 ± 9.35

TABLE V

Averaged over 15 trials for three robots and the noise was set to 5%.

	No Partition	Voronoi
1 Hotspot	4.73 ± 2.15	4.86 ± 1.88
2 Hotspots	31.46 ± 5.73	11.93 ± 4.00
3 Hotspots	50.06 ± 12.69	20.53 ± 8.81
4 Hotspots	71.13 ± 15.00	44.33 ± 9.80

- 1) Boustrophedon (BST): Every robot individually follows a boustrophedon pattern.
- No partition: Robots can explore the entire environment anytime without being restricted to their Voronoi partition.
- 3) Site partition: Robots are limited to their Voronoi partitions, determined by their last surfacing event.

We compare the Voronoi partitioning and No partitioning in terms of the time taken by them to find all the hotspots (4 for the synthetic field). Table IV shows the earliest time for four robots to detect 1, 2, 3, and 4 hotspots. The Voronoi partitioning achieves better exploration and outperforms No Partitioning when it comes to finding multiple hotspots. Table V shows the earliest time for three robots to detect 1, 2, 3, and 4 hotspots.

E. Chlorophyll Dataset

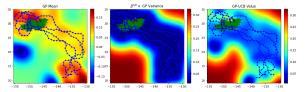
We run ten missions for four robots that start with starting locations (-135, 12), (-132, 12), (-137, 12), (-138, 11) 5. The selected locations near the local maxima divert robots from the hotspot, encouraging them to explore more broadly. Comparing scenarios with and without partitioning shows that partitioning facilitates more uniform exploration and lowers GP variance, as seen in Figure 8 after 50 time units. Without it, robots often cover the same areas, leading to redundant measurements. Table VI presents metrics analogous to Table I and Figure 9 mirrors Figure 6. The two Voronoi-based methods outperform the Boustrophedon pattern (represented by the red plot). Utilizing Voronoi partitioning offers a distinct edge over not using it (Green plot). Without partitioning, robots risk redundant measurements in overlapping areas. Voronoi partitioning efficiently distributes exploration among robots.

VI. CONCLUSION

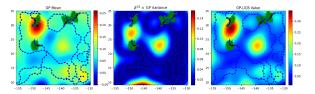
In this study, we investigated hotspot identification using both single and multiple mobile sensors. We introduced the AdaptGP-MCTS algorithm designed for scenarios with unknown GP hyperparameters. While adaptive hyperparameter optimization is computationally intensive, our results indicate that AdaptGP-MCTS is a viable option for users with ample

TABLE VI
The time budget is 130 units.

BST	No Partition	Voronoi
20.20 ± 1.91	14.10 ± 9.43	7.34 ± 2.81
77.71 ± 0.52	72.81 ± 1.60	72.78 ± 1.59
40.14 ± 0.86	22.20 ± 1.42	18.21 ± 0.67
67.54 ± 9.49	30.81 ± 13.71	16.57 ± 1.68



(a) No partitioning path after the robots have spent 50 units of time.



(b) Voronoi partitioning path after the robots have spent 50 units of time.

Fig. 8. Voronoi partitioning vs No partitioning comparison at time 50.

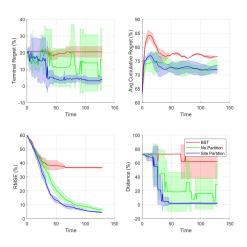


Fig. 9. The sensor noise standard deviation was set to 5%.

mission time but limited computing power on their robotic system. For shorter missions, when the kernel's prior hyperparameters are unknown, OptGP-MCTS emerges as the recommended choice. For multiple robots, we employed a Voronoi region-based partitioning system to delegate specific environment subregions to individual robots. Path planning within these partitions harnesses the UCB values from the learned GP model, enabling robots to strike a balance between exploration and exploitation. As a direction for future research, it would be intriguing to assess the performance of an AdaptGP-MCTS-style planner in multi-robot scenarios.

REFERENCES

- P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic UAV and UGV system for precision agriculture," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1498–1511, 2016.
- [2] V. Suryan and P. Tokekar, "Learning a spatial field in minimum time with a team of robots," *IEEE Transactions on Robotics (TRO)*, vol. 36, no. 5, pp. 1562–1576, Oct. 2020.
- [3] W. Aktar, D. Sengupta, and A. Chowdhury, "Impact of pesticides use in agriculture: their benefits and hazards," *Interdisciplinary toxicology*, vol. 2, no. 1, pp. 1–12, 2009.
- [4] A. Blanchard and T. Sapsis, "Informative path planning for anomaly detection in environment exploration and monitoring," arXiv preprint arXiv:2005.10040, 2020.
- [5] Y. Sung, D. Dixit, and P. Tokekar, "Online multi-robot exploration of a translating plume: Competitive algorithm and experiments," arXiv preprint arXiv:1811.02769, 2018.
- [6] T. Moore, "In the usa, robotics technology used at the tmi-2 cleanup and at other nuclear plants has prompted interest and shaped research on how robots might best be used," *IAEA BULLETIN*, p. 31, 1985.
- [7] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012.
- [8] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," arXiv preprint arXiv:0912.3995, 2009.
- [9] M. Tanemura, T. Ogawa, and N. Ogita, "A new algorithm for threedimensional voronoi tessellation," *Journal of Computational Physics*, vol. 51, no. 2, pp. 191–207, 1983.
- [10] Y. Teck Tan, A. Kunapareddy, and M. Kobilarov, "Gaussian process adaptive sampling using the cross-entropy method for environmental sensing and monitoring," in *International Conference on Robotics and Automation* 2018, 05 2018, pp. 6220–6227.
- [11] C. Mellucci, P. P. Menon, C. Edwards, and P. Challenor, "Source seeking using a single autonomous vehicle," in 2016 American Control Conference (ACC), 2016, pp. 6441–6446.
- [12] E. Rolf, D. Fridovich-Keil, M. Simchowitz, B. Recht, and C. J. Tomlin, "A successive-elimination approach to adaptive robotic sensing," *CoRR*, vol. abs/1809.10611, 2018. [Online]. Available: http://arxiv.org/abs/1809.10611
- [13] W. Chen and L. Liu, "Pareto monte carlo tree search for multiobjective informative planning," in *Proceedings of Robotics: Science* and Systems, FreiburgimBreisgau, Germany, June 2019.
- [14] A. Singh, A. Krause, C. Guestrin, W. Kaiser, and M. Batalin, "Efficient planning of informative paths for multiple robots," in *Proceedings of* the 20th International Joint Conference on Artifical Intelligence, ser. IJCAI'07. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, p. 2204–2211.
- [15] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, no. Feb, pp. 235–284, 2008.
- [16] J. Binney, A. Krause, and G. S. Sukhatme, "Optimizing waypoints for monitoring spatiotemporal phenomena," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 873–888, 2013.
- [17] V. Suryan and P. Tokekar, "Learning a spatial field with gaussian process regression in minimum time," in *Algorithmic Foundations of Robotics XIII*. Cham: Springer International Publishing, 2020, pp. 301–317.
- [18] S. Kemna, O. Kroemer, and G. S. Sukhatme, "Pilot surveys for adaptive informative sampling," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 6417–6424.
- [19] C. Xiao and J. Wachs, "Nonmyopic informative path planning based on global kriging variance minimization," *IEEE Robotics and Automa*tion Letters, 2022.
- [20] S. Gelly, L. Kocsis, M. Schoenauer, M. Sebag, D. Silver, C. Szepesvári, and O. Teytaud, "The grand challenge of computer go: Monte carlo tree search and extensions," *Communications of the ACM*, vol. 55, no. 3, pp. 106–113, 2012.
- [21] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012

- [22] R. Marchant, F. Ramos, and S. Sanner, "Sequential bayesian optimisation for spatial-temporal monitoring," in *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'14. Arlington, Virginia, USA: AUAI Press, 2014, p. 553–562.
- [23] D. Tolpin and S. Shimony, "Mcts based on simple regret," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 26, no. 1, 2012, pp. 570–576.
- [24] A. Barrientos, J. Colorado, J. d. Cerro, A. Martinez, C. Rossi, D. Sanz, and J. Valente, "Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots," *Journal of Field Robotics*, vol. 28, no. 5, pp. 667–689, 2011.
- [25] W. Kazmi, M. Bisgaard, F. Garcia-Ruiz, K. D. Hansen, and A. la Cour-Harbo, "Adaptive surveying and early treatment of crops with a team of autonomous vehicles," in *Proceedings of the 5th European Conference on Mobile Robots ECMR* 2011, 2011, pp. 253–258.
- [26] A. Dhariwal, G. S. Sukhatme, and A. A. Requicha, "Bacterium-inspired robots for environmental monitoring," in *IEEE International Conference on Robotics and Automation*, 2004. Proceedings. ICRA'04. 2004, vol. 2. IEEE, 2004, pp. 1436–1443.
- [27] M. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 24–39, 2012.
- [28] M. Ouimet and J. Cortés, "Collective estimation of ocean nonlinear internal waves using robotic underwater drifters," *IEEE Access*, vol. 1, pp. 418–427, 2013.
- [29] G. P. Kontoudis and D. J. Stilwell, "Fully decentralized, scalable gaussian processes for multi-agent federated learning," 2022. [Online]. Available: https://arxiv.org/abs/2203.02865
- [30] R. Zlot, A. Stentz, M. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," in *Proceedings 2002 IEEE Interna*tional Conference on Robotics and Automation (Cat. No.02CH37292), vol. 3, 2002, pp. 3016–3023 vol.3.
- [31] C. Nieto-Granda, I. John G. Rogers, and H. I. Christensen, "Coordination strategies for multi-robot exploration and mapping," *The International Journal of Robotics Research*, vol. 33, no. 4, pp. 519–533, 2014. [Online]. Available: https://doi.org/10.1177/ 0278364913515309
- [32] J. Yuan, Y. Huang, T. Tao, and F. Sun, "A cooperative approach for multi-robot area exploration," in 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, pp. 1390–1395.
- [33] D. E. Soltero, M. Schwager, and D. Rus, "Generating informative paths for persistent sensing in unknown environments," in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 2172–2179
- [34] A. Marino, G. Antonelli, A. P. Aguiar, A. Pascoal, and S. Chiaverini, "A decentralized strategy for multirobot sampling/patrolling: Theory and experiments," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 1, pp. 313–322, 2015.
- [35] S. Kemna, J. G. Rogers, C. Nieto-Granda, S. Young, and G. S. Sukhatme, "Multi-robot coordination through dynamic voronoi partitioning for informative adaptive sampling in communication-constrained environments," in 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 2124–2130.
- [36] W. Luo and K. Sycara, "Adaptive sampling and online learning in multi-robot sensor coverage with mixture of gaussian processes," in 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 6359–6364.
- [37] F. Berkenkamp, A. P. Schoellig, and A. Krause, "No-regret bayesian optimization with unknown hyperparameters," arXiv preprint arXiv:1901.03357, 2019.