Fully Autonomous Focused Exploration for Robotic Environmental Monitoring

Gregory Hitz[1], Alkis Gotovos[2], François Pomerleau[1], Marie-Ève Garneau[3], Cédric Pradalier[4], Andreas Krause[2] and Roland Y. Siegwart[1]

***Abstract*— Robotic sensors are promising instruments for monitoring spatial phenomena. Oftentimes, rather than aiming to achieve low prediction error everywhere, one is interested in determining whether the phenomenon exhibits certain critical behavior. In this paper, we consider the problem of focusing autonomous sampling to determine whether and where the sensed spatial field exceeds a given threshold value. We introduce a receding horizon path planner, LSE-DP, which plans efficient paths for sensing in order to reduce our uncertainty specifically around the threshold value. We report fully autonomous field experiments with an Autonomous Surface Vessel (ASV) in an aquatic monitoring setting, which demonstrate the effectiveness of the proposed method. LSE-DP is able to reduce the uncertainty around the threshold value of interest to 68% when compared to non-adaptive methods.**

I. Introduction

The analysis of both temporal and spatial dynamics of environmental phenomena requires large data sets, which have to be collected in potentially harsh conditions. Recent advances in robotics facilitate the addition of mobility to sensors and thus allow to collect data very specifically at desired locations. The selection of measurement sites is a difficult problem, which has been addressed in recent research. In the simplest setting, the objective of such a selection process is to get uniformly high confidence about the observed process over a given area. However, in many cases it might be preferable to set a *specific focus on particular aspects of the phenomenon, for example, where it crosses a certain critical threshold value*. With the targeted investigation of a threshold value, the observed phenomenon can be classified into areas that are lower or higher than the threshold. In this paper we investigate this problem, propose a solution and apply it to a real environmental monitoring application. Handling large data sets is a problem in biology [1], which suggests that focusing on interesting areas may lead to more efficient analysis of complex ecosystems.

In particular, our work aims at monitoring level sets within the spatial distribution of cyanobacteria in a lake. Concretely, we study the distribution of *Planktothrix rubescens* in Lake Zurich, which we sense using an Autonomous Surface Vessel (ASV). *P. rubescens* is a toxic cyanobacteria, which is common in pre-alpine lakes and usually blooms in late summer and fall. Optimal conditions for *P. rubescens* growth are a vertically stable water column and a low light regime. These are met within the metalimnion (stable layer of highest temperature gradients), which is localized between 5 and 18 m depth in Lake Zurich [1]. Therefore, *P. rubescens* forms a horizontal layer during summer and fall. A 40-year study of its seasonal abundance in Lake Zurich has shown that the biovolume of this toxic cyanobacteria has increased significantly over the last decades [2], emphasizing the importance of regularly collecting high-density data sets. We collect measurements with the ASV *Lizhbeth* [3]. The robot is equipped with a winch system that allows the user to lower an aquatic sensor unit into the water (YSI 6600, hereinafter referred to as *probe*), which measures the fluorescence of phycoerythrin, the main pigment of *P. rubescens*, in relative fluorescence units (RFU). The winch allows the robot to take measurements down to a depth of 25 m while moving. The cylindrical probe is carried in a custom, torpedo-like support structure (see Figure 1), to minimize drag and ensure a stable lateral position. Further details about the robotic system and its applications are provided by Hitz et al. [3].

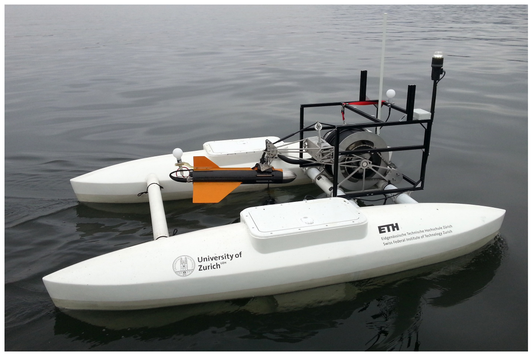

Fig. 1. The Autonomous Surface Vessel on Lake Zurich on a calm day. The YSI probe is mounted inside the black and yellow torpedo situated in between the hulls and is ready to be lowered into the water.

A. Related work

The review of Dunbabin and Marques [4] provides an extensive overview of applications of robots as mobile sensing agents. Whereas some applications build on the passive circulation of agents (such as the Argo floats [5]), the majority of works use active agents and aim at planning optimal paths to retrieve the most useful measurements from the environment. Krause et al. [6] discuss the sensor placement problem, which consists of selecting the most informative sites for a given number of sensors. This is a related problem but it is slightly simpler as the selected sites

[1] Autonomous Systems Lab – ETH Zurich, Switzerland
[2] Learning & Adaptive Systems Group – ETH Zurich, Switzerland
[3] Limnological Station Kilchberg – University of Zurich, Switzerland
[4] DREAM Lab – GeorgiaTech Lorraine, France

for sensors do not have to be connected to a feasible and efficient path.

Aquatic monitoring projects, like ours, often deploy submersible robots (autonomous underwater vehicles, AUV). For such systems, Binney et al. [7] describe a path planning algorithm with the aim of reducing the overall uncertainty of the scalar field of interest. In such a scenario, the path can be planned offline, since the actual values of the measurements do not influence the planning. Smith et al. [8] discuss the problem of optimizing measurement resolutions while considering ocean currents when designing coverage paths for AUVs. Hollinger and Singh [9] describe an algorithm to plan paths for multiple agents for searching a target in a known environment. Their approach is similar to ours as they also make use of a receding horizon planner to overcome computational complexity. Decentralized methods to deploy multiple robots for exploring environmental processes are discussed by Low et al. [10]. In a different context, Hollinger et al. [11] discuss the benefit of adaptively planning measurements for robotic underwater ship inspection, which is similar in the sense that the inspection robot needs to adapt its path online. A data-adaptive planner targeted at the estimation of boundaries is presented by Singh et al. [12]. Their work is similar to ours, however, in their approach they separate the problem in a coarse coverage run and a refinement run. Gotovos et al. [13] present the *Level Set Estimation (LSE)* algorithm which specifically selects sampling sites to estimate level sets. However they only briefly discuss the problem of connecting the selected sites to a path.

When doing robotic field experiments in a dynamic environment, one difficulty arising is the missing ground truth to compare results against. The NIMS project, which aimed originally at observing light intensity within a forest canopy [14] and aquatic environments [15], deals with that challenge by generating static lighting conditions in a laboratory environment in order to create dense ground truth measurements. We reused the same methodology by doing coverage runs before and after our experiments.

B. Contributions

In this paper, we propose an approach towards fully autonomous robotic monitoring of critical thresholds in environmental phenomena. Our approach builds on the LSE algorithm [13] and extends it with a path planner that constrains the selected sites to lie on a feasible path for a robot. In particular, we use a dynamic programming approach with a receding horizon to plan a sampling path for the probe within a predefined vertical transect plane. The core contributions of this paper are:

1) A receding horizon path planner for the LSE algorithm for choosing measurement sites on an efficient path.
2) A performance evaluation of the path planner in simulation and a discussion of reasonable parameters.
3) Results from fully autonomously executed field tests on a lake, for which we also recorded comparative data sets on a uniform grid.

II. Problem statement and background

We seek to infer knowledge about an unknown scalar field $f : \mathbb{R}^d \mapsto \mathbb{R}$ (in this application the distribution of *P. rubescens*) from samples $\mathcal{Y}_t = \{y_1, \ldots, y_t\}$ taken at locations $\mathcal{A}_t = \{x_1, \ldots, x_t\}$, selected from a finite set of potential measurement sites $\mathcal{D} \subseteq \mathbb{R}^d$. Given a constant threshold h, we seek to classify all points $x \in \mathcal{D}$ into either a superlevel set $\mathcal{H} = \{x \,|\, f(x) > h\}$ or a sublevel set $\mathcal{L} = \{x \,|\, f(x) < h\}$. Besides selecting sites x_i at which we require measurements y_i, we want to visit these locations along a path $\mathcal{P} = \langle x_1, \ldots, x_t \rangle$, ideally as short as possible, which obeys the motion constraints of the robot.

A. Gaussian process

Gaussian processes (GP)[1] offer a principled way to model the unknown field f non-parametrically, while allowing to encode certain assumptions about the smoothness of f in the form of hyper-parameters. A mean function $m(\boldsymbol{x})$ and a covariance function (or kernel) $k(\boldsymbol{x}, \boldsymbol{x}')$ fully specify a Gaussian process $\mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$ [17]. Whereas the mean function formulates prior knowledge about the values of $f(\boldsymbol{x})$, the kernel function encodes the smoothness of f. Under the assumption of constant Gaussian measurement noise with zero mean ($y_i = f(x_i) + e_i$ where $e_i \sim \mathcal{N}(0, \sigma^2)$), we can formulate the mean, covariance matrix and variance of the posterior over f as follows [17]:

$$\mu_t(\boldsymbol{x}) = k_t(\boldsymbol{x})^T \left(\boldsymbol{K}_t + \sigma^2 \boldsymbol{I}\right)^{-1} \mathcal{Y}_t \tag{1}$$

$$k_t(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}_t(\boldsymbol{x})^T \left(\boldsymbol{K}_t + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}_t(\boldsymbol{x}) \tag{2}$$

$$\sigma_t^2(\boldsymbol{x}) = k_t(\boldsymbol{x}, \boldsymbol{x}) \tag{3}$$

$$\text{where } \boldsymbol{k}_t(\boldsymbol{x}) = [k(x_1, \boldsymbol{x}), \ldots, k(x_t, \boldsymbol{x})]^T$$
$$\text{and } \boldsymbol{K}_t = [k(x, x')]_{x, x' \in \mathcal{A}_t}$$

B. Level set estimation

Given a set of noisy measurements $\mathcal{Y}_t$ at locations $\mathcal{A}_t$, we want to classify the set $\mathcal{D}$ into the two sets $\mathcal{L}_t$ (*low*) and $\mathcal{H}_t$ (*high*) which are defined by the threshold value h. To this end, we use the GP posterior mean $\mu_t(\boldsymbol{x})$ and variance $\sigma_t(\boldsymbol{x})$ and take the confidence bounds introduced by Gotovos et. al. [13] into account:

$$\mathcal{H}_t = \{x \,|\, \mu_t(x) - \beta\, \sigma_t(x) > h\} \tag{4}$$

$$\mathcal{L}_t = \{x \,|\, \mu_t(x) + \beta\, \sigma_t(x) < h\} \tag{5}$$

Using the predictive variance of the GP ensures that we only classify points x for which f is higher or lower than the threshold with high certainty. The parameter β allows to control the level of required confidence for classification. In general, not all $x \in \mathcal{D}$ can be classified into $\mathcal{L}_t$ or $\mathcal{H}_t$ leaving a set of remaining uncertain points:

$$\mathcal{U}_t = \mathcal{D} \setminus (\mathcal{L}_t \cup \mathcal{H}_t). \tag{6}$$

Our goal is to efficiently gain enough information about the points in $\mathcal{U}_t$, to classify them according to Equations 4 or

[1]also called *Kriging models* in Geostatistics, where they are often used to model spatial phenomena [16].

5. The LSE algorithm [13] defines the measure of *ambiguity* for each unclassified point to assess the need for taking a sample at its location. Gotovos et al. [13] also discuss different measures such as *mutual information*. In addition to the basic LSE algorithm, which selects single points sequentially, they also discuss the batch processing version for the case where multiple points have to be selected in a single iteration. They apply a naive path planning scheme using a Traveling Salesman Problem (TSP) formulation, which connects the measurement sites selected in a batch. This is related to our approach in the sense that the expected measurements along a path are evaluated jointly in a batch. However in the TSP version the resulting paths in each iteration are optimal (depending on the applied TSP heuristic) but the global path resulting from connecting paths of sequentially selected batches can still be quite inefficient. This motivates the development of our receding horizon planner.

III. Path planning

To plan a path within some operation space of dimension d (usually 2D or 3D), we set up a graph $\mathcal{G}(V, E)$ with a set of vertices V and edges E. In general, any graph could be used. By adding constraints to its structure, motion limitations of the robot and application-based restrictions can be encoded. We discuss specifics of our implementation in Section IV-A.

To compare different measurement paths, a metric is required to assess the "usefulness" of a path $\mathcal{P}$ with respect to the already available measurements. We use the conditional mutual information of a set A of equally spaced measurements along $\mathcal{P}$ with respect to the already available measurements $\mathcal{Y}_t$.

$$F_t(A) = \mathrm{MI}(f; y_A | y_t) = \frac{1}{2}\log\left(\left|I + \sigma^{-2}\boldsymbol{K}_A^t\right|\right) \tag{7}$$

The correlation of y_A and y_t is defined by Equation 2: $\boldsymbol{K}_A^t = k_t(x, x')_{x,x' \in A}$. Since we want to gain information about the unclassified areas, we only take unclassified points into consideration:

$$\bar{F}_t(A) = F_t(\bar{A}) = \mathrm{MI}(f; y_{\bar{A}} | y_t) \quad \text{with} \quad \bar{A} = A \cap \mathcal{U}_t \tag{8}$$

We use a dynamic programming approach with which we plan a path for a receding horizon of fixed size h_{dp}. At iteration t, this results in a path $\mathcal{P}_t = \langle x_{t,1}, \dots, x_{t,h_{dp}} \rangle$ of which the robot executes the first segment from $x_{t,1}$ to $x_{t,2}$. After that, we re-plan the path with $x_{t+1,1} = x_{t,2}$. During each iteration t, the LSE-DP[2] algorithm loops over h_{dp} sets of vertices $\{V_k\}_{k=1,\dots,h_{dp}}$. The first set only consists of the starting point x_s and the following are built according to Equation 9. They contain all vertices that are connected to at least one of the vertices in the previous set. Here, we assume that all edges have equal traveling cost, for reasons of clarity[3]. The sets V_k are shown schematically in Figure 2. For each vertex x_i we initialize the best path $p_i^\star$ to the starting point $x_{t,1}$ as an empty sequence.

$$\begin{aligned} V_1 &= \{x_{t,1}\} \\ V_k &= \{x \in \mathcal{D} \mid (x', x) \in E \ \wedge \ x' \in V_{k-1}\} \\ p_i^\star &= \langle\rangle \end{aligned} \tag{9}$$

We process the sets of vertices sequentially in a forward direction. Starting at $k = 2$, we optimize paths for each vertex x of V_k by evaluating the combinations of the edges $e_{x',x}$ to all possible predecessor x' and their best paths $p_{x'}^\star$:

$$p_x^\star = \underset{p \in P_{x',x}}{\operatorname{argmax}} \left(\bar{F}_t(p)\right) \tag{10}$$

$$P_{x',x} = \left\{\langle p_{x'}^\star, e_{x',x}\rangle \mid e_{x',x} \in E\right\}, \quad x' \in V_{k-1}, \quad x \in V_k$$

Figure 2 shows an example of this procedure schematically. The part highlighted in blue is processed already, making V_3 the set of vertices that is currently processed. The vertex x_4 is currently considered. The best path $p_4^\star$ is the maximizer of the mutual information along the three potential paths: $\langle p_1^\star, e_{1,4}\rangle$, $\langle p_2^\star, e_{2,4}\rangle$ and $\langle p_3^\star, e_{3,4}\rangle$. This procedure is very similar to the Dijkstra algorithm [18]. However, we have to take into account the *diminishing returns* property of the conditional mutual information, which we use as an evaluation metric: the "usefulness" of an edge is not constant, but depends on the other parts of the path. Thus we can not simply add up the gains of separate edges, but have to re-evaluate the entire path back to the initial vertex x_s for the evaluation at each vertex. The diminishing returns property is a consequence of the *submodularity* of the mutual information function and results in the following inequality for our example: $\bar{F}_t(p_1^\star) + \bar{F}_t(e_{1,4}) \geq \bar{F}_t(\langle p_1^\star, e_{1,4}\rangle)$.

During the initial setup of the graph, no information about the boundaries of the level sets is available, meaning that the initial setup of the graph might not give the desired resolution in the unclassified regions (after a certain number of iterations when $|\mathcal{U}_t|$ is relatively small). For this reason, a predefined number of vertices can be added to the graph, which lie in the unclassified regions. This can be done in multiple ways. We discuss the method we implemented in section IV-A.

The complete algorithm: Algorithm 1 describes the overall procedure, which repeatedly plans a path and executes the first step until all points in $\mathcal{D}$ are classified. In practice, different stopping criteria might be required, for example, limiting the number of iterations, or the maximal number of remaining unclassified points.

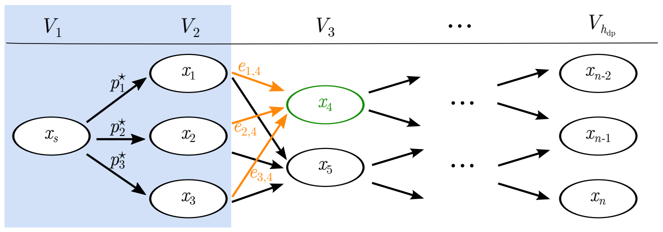

Fig. 2. Schema of the planning procedure. The sets V_k, considered sequentially, are arranged in columns. The vertices x_i are processed in sequence.

[2]LSE-DP stands for Level Set Estimation with Dynamic Programming
[3]Our approach naturally extends to non-uniform edge lengths.

Algorithm 1 The LSE-DP algorithm

```
Input: Graph $G$, planning horizon $h_{dp}$, starting vertex $x_0$
Output: level sets $H$ and $L$
1: $L_0 = \emptyset, H_0 = \emptyset, U = D$
2: $t \leftarrow 1$
3: $x_s \leftarrow x_0$
4: while $U_{t-1} \neq \emptyset$ do
5:     $G_t \leftarrow$ add_adaptive_vertices($G, U_t$)
6:     $p_t^\star \leftarrow$ plan_path($G_t, x_s, h_{dp}$)
7:     $y_t \leftarrow$ sample_along($\langle x_s, p_{t,1}^\star \rangle$)
8:     $H_t, L_t, U_t \leftarrow$ classify($y_{1:t}$)
9:     $x_s \leftarrow p_{t,1}^\star, \quad t \leftarrow t + 1$
```

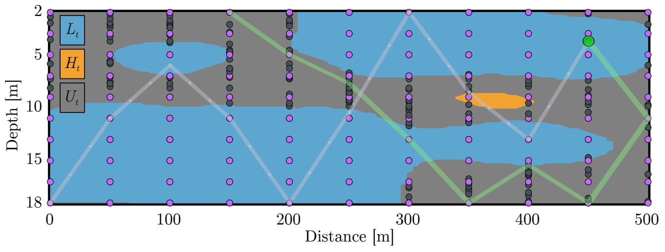

Fig. 3. The gray path shows the path that was previously executed, along with the positions of samples (gray dots). The pink circles depict the positions of the vertices and the gray circles the ones that are adaptively added. The green line shows the planned path. The background is colored according to the classes *low* ($\mathcal{L}_t$), *high* ($\mathcal{H}_t$) and *unclassified* ($\mathcal{U}_t$).

IV. Experiment setup

A. Implementation details

a) GP model details: For our application, we have chosen an *anisotropic squared exponential kernel* of the following form:

$$k(x, x') = \sigma_f \exp\left(-(x - x')^T L \, (x - x')\right), \tag{11}$$

where L is a diagonal matrix with diagonal elements $(1/l_1, \ldots, 1/l_d)$ and l_i is the length scale parameter for the i-th dimension. The anisotropicity of this kernel function accounts for the large differences in spatial dynamics that the problem at hand exhibits. Furthermore, we used a constant mean function $m(x) = m_{gp}$. Since we operate in a two-dimensional space, this yields a set of five hyper-parameters: $[m_{gp}, \sigma_f, l_1, l_2, \sigma]$, which were optimized with respect to the marginal likelihood based on data sets from an earlier sampling campaign (during summer 2011, [1]).

b) Motion constraints: The robot operates in a two-dimensional space and uses two different actuators to control the position along each of the dimensions. The boat is controlled along a predefined line on the lake with constant speed. The maximal power of the motors and the minimal speed that can be measured by differentiating GPS measurements define the range $[\bar{v}_{h,\text{min}}, \bar{v}_{h,\text{max}}]$ of applicable speed $\bar{v}_h$ along the horizontal axis $\vec{e}_h$. The vertical motion of the probe is controlled by the winch for which the power of the motor imposes limits on the maximal vertical speed: $\bar{v}_v \in [-\bar{v}_{v,\text{max}}, \bar{v}_{v,\text{max}}]$. We translate these velocity constraints to connectivity constraints in the graph by defining a range for the inclination that an edge can have: $m_e \in [-\bar{v}_{v,\text{max}}/\bar{v}_{h,\text{min}}, \bar{v}_{v,\text{max}}/\bar{v}_{h,\text{min}}]$.

To minimize the energy consumption of the robot, we impose further constraints. Accelerating the boat, stopping and turning around are energetically expensive operations. Furthermore, a constant horizontal speed aids the control of the depth of the probe. Since the probe is tethered via a cable, the speed of the boat influences the depth due to drag and lift forces (further details in [3]). Therefore, we constrain the speed of the boat to a constant value: $\bar{v}_h = \bar{v}_{h,\text{set}}$ and minimize the number of stops by only allowing a change of direction at the end of a transect.

c) Size constraints: We organize the vertices in the graph in constantly spaced columns along the horizontal axis. The example shown in Figure 3 illustrates the columns of vertices as pink circles. Connections between vertices are only allowed among neighboring columns and due to the constraints on the inclination of edges m_e, vertices of the same column can not be connected. There remain two parameters for the density of the vertices: the horizontal and vertical spacing of the nodes (s_h and s_v, respectively). The controller of the boat uses a circle of 3 m radius around the target point to detect the arrival at the target. Combined with the positioning accuracy of the GPS receiver ($\sim$0.5 - 2 m, depending on the speed of the boat) yields a theoretical minimal value for s_h of 5 m.

d) Adaptive vertices: As described above, we adaptively add vertices in each iteration to give the unclassified areas more chances to be visited. We select their depth from a uniform distribution over the entire height of the column and keep them if they are in $\mathcal{U}_t$. In the example of Figure 3, these additional vertices are shown as gray circles.

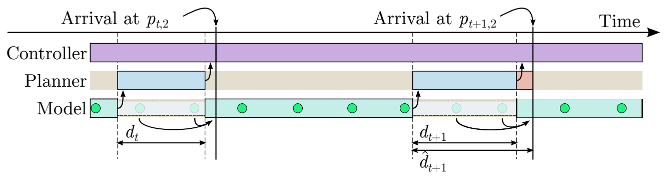

Fig. 4. Schematic timeline. Planning is executed early enough to be completed *before* arriving at the next vertex. During this time, new measurements (green circles) are buffered and only added to the model after planning has completed.

e) Real-time implementation on the robot: With an increasing number of samples, the planning iterations with LSE-DP require more time. This has to be considered for applying the planner in a real-time application on the lake. Once the boat arrives at a vertex, it would be inefficient in terms of time and energy to stop and wait for the new path to be computed. Therefore, we want to finish the new planning iteration by the time that we arrive at the next vertex, such that the boat can directly continue with the updated path. Since the boat is traveling at constant speed, we can estimate the distance from the vertex at which we have to start the planning. Thus, given the duration d_t that the planner took to compute the path at iteration t, we predict the duration of the next planning step by an incremental linear factor: $\hat{d}_{t+1} = k \cdot d_t$, with $k \geq 1$. Assuming an exponential increase is over-conservative, but ensures that the planning algorithm finishes in time. Figure 4 shows a schematic representation of the real-time implementation along a timeline. Once the boat is close enough to the next vertex (closer than $\hat{d}_{t+1} \cdot v_{h,\text{set}}$), the new planning iteration is started. While a new path is computed no measurements can be added to the model. We buffer them and add them once the planning procedure is completed.

B. Simulation experiments

We ran simulations on a data set that was previously acquired in the lake. As dense ground truth, we considered a GP posterior mean, conditioned on the real measurements. In simulation, we compared our algorithm to three other sequential methods to plan a path on the same graph.

1) **Random**: As a baseline for comparison, we implemented a random selection scheme, which randomly selects one of the connected vertices of the next column at each iteration.
2) **Greedy $\mathbf{F_t}$**: This planner uses mutual information over all points as described in Equation 7. It does not use a look ahead, meaning that it greedily plans one step at a time.
3) **Greedy $\mathbf{\bar{F}_t}$** This method uses the mutual information of only unclassified points (Equation 8), but also plans greedily. This is the LSE-DP planner with a horizon of 1.

The planning horizon for the LSE-DP planner was set to 6. Larger horizons did not show significant improvements in our experiments. As the exact ground truth of the data was available, we used the F_1 score to evaluate the quality of the classification on a dense grid of 10'000 points. Unclassified points counted as wrongly classified.

C. Field tests

The experiments were conducted on Lake Zurich during August 2013. Figure 5 shows the location of the experiment transect, which is 500 m long. We selected the length of the transect such that the robot can cover it up to 12 times consecutively. This enabled us to collect data on a uniform grid before and after the experiment, providing a comparison data set for the evaluation of the adaptively planned path. Thus, the experiment procedure consisted of three parts: (1) Coverage on an uniform grid, (2) the adaptive level set estimation experiment using the LSE-DP algorithm, (3) a second coverage run along the same trajectory as the first one. We assume that the temporal dynamics of *P. rubescens* are slow enough to be neglected. Taking coverage data sets before and after the adaptive run allowed us to validate this assumption.

The experiments were executed fully autonomously by the robot using the real-time strategy described above. In total, we have carried out three experiments on different days. Besides the initial configuration of the parameters, no external input was given to the robot during the entire run of the experiment. Table I gives an overview of the parameters that were used during the experiments. The length scale parameters of the *P. rubescens* distribution are quite stable over the summer and the measurement noise is mainly depending on the probe, which is the same for all experiments. The constant mean value m_{gp} and the threshold value h have to be set according to the current density of *P. rubescens* cells. Appropriate values were chosen based on previous experimental work [1]. The planning horizon was chosen higher than in the simulation experiments described above as the expected computation times allow this slightly more conservative choice.

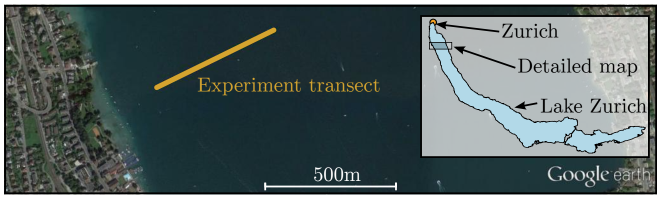

Fig. 5. Map of the testing area. The testing transect is indicated in orange.

TABLE I
OVERVIEW OF PARAMETERS

	Parameter	Value	Description
Model	l_1	148.41 m	Horizontal length scale of the GP kernel
	l_2	1.57 m	Vertical length scale of the GP kernel
	σ_f	4.06 RFU	Signal noise of the GP kernel
	σ	1.1 RFU	Measurement noise of the GP kernel
	m_{gp}	4.4 RFU	Mean function of the GP
Graph	$\bar{v}_{h,\max}$	1.0 m/s	Maximal boat speed
	$\bar{v}_{h,\text{set}}$	0.6 m/s	Target boat speed
	$\bar{v}_{v,\max}$	0.1 m/s	Maximal winch speed
	s_h	55.5 m	Horizontal resolution of $\mathcal{G}$
	s_v	1.77 m	Vertical resolution of $\mathcal{G}$
Planning	n_e	10	Number of samples per edge
	n_v	15	Number of adaptive vertices per column
	h	8 RFU	Threshold value for classification
	h_{dp}	8	Planning horizon

V. RESULTS

A. Simulation experiments

Figure 6 compares the performance of the different methods we implemented to plan sampling paths on the graph. The left plot shows the median performance of each method against the number of samples. On the right, the worst case scenario is shown. Clearly and expectedly, the random version performed worst. The fact that also the random planner eventually achieves relatively good results after adding many samples is due to the trellis structure of the graph, which enforces it to traverse the entire transect. Furthermore, the adaptive addition of vertices increases the chances of randomly selecting one in the unclassified region. Both greedy planners achieve relatively good results on the long run. Again the structure of the graph works in their favor. However, both greedy planners have specific disadvantages when compared to the forward looking planner (LSE-DP): The first greedy planner (greedy F_t) takes all points into consideration and thus quickly gains information about the entire distribution evenly. With increasing numbers of samples, however, it does not focus on the boundary areas (by design) and the performance plateaus. The second greedy planner (greedy $\bar{F}_t$) only focuses on unclassified points. If no point in the unclassified set can be reached in one step, the next point has to be chosen blindly. This leads to similar behavior as the random planner and the worst case shows for instance that it requires almost three times as many samples as the LSE-DP algorithm for a score of 0.6. The LSE-DP planner shows the best performance for two reasons: 1) its score rises quickly, which corresponds to an efficient classification of the majority of the points, 2) it focuses best on the unclassified regions, resulting in the highest final score.

B. Field experiments

In total, we have conducted three experiments and, for each of them, we have conducted a coverage run before and after the LSE-DP experiment. Figure 7 shows the results from one of the experiments. The plot on the top shows the posterior mean of the GP model inferred from the first coverage run. The cyanobacterial layer was very stratified, without a clear maximum along the horizontal direction. This provided a rather simple instantiation of the problem, since only two horizontal boundaries were present. However, situations of thermal stratification are common in aquatic ecosystems and our data sets are thus quite representative. The other two experiments have similar distributions. The second plot shows the classification resulting from the first coverage run without taking into account the variance-based confidence bounds (hence all points are classified). The last plot shows the adaptively planned path with the resulting classification. The second coverage run was done after the adaptive run, but is not shown in Figure 7 as it is very similar.

The two coverage runs before and after the LSE-DP experiment can be used to assess the temporal stability of the field. In order to compare the two coverage runs, we evaluated the GP mean on a dense grid of 10'000 points

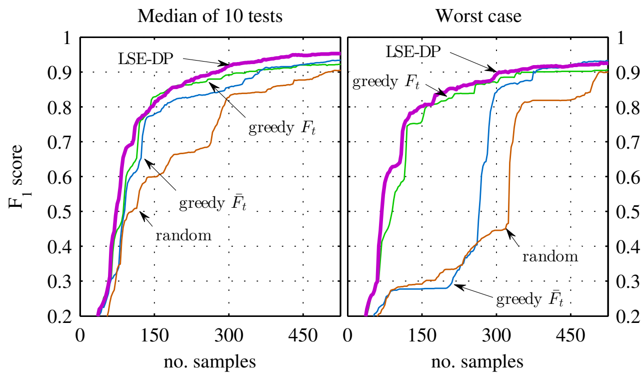

Fig. 6. Comparison of simulation results for different planning methods.

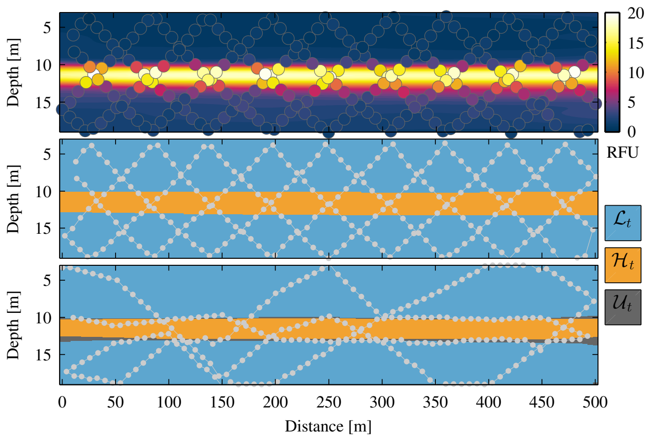

Fig. 7. Results from the field experiment on Lake Zurich. *Top*: Samples of the first coverage run with the mean of the GP posterior with the color representing *P. rubescens* level in RFU. *Middle*: The resulting ground truth classification with the path of the probe in gray. *Bottom*: Resulting path from the LSE-DP experiment.

for each run separately. If there were no temporal changes, the difference of the posterior means should be very small (i. e., only depend on measurement noise). For the three experiments, these differences yielded the following RMS values relative to the maximal measurement value: 8.3%, 3.2% and 2.9%. This indicates temporally stable conditions for experiment two and three. In the first experiment the error value is higher. To assess if this had an influence on the location of the threshold value, we classified the field based both on the first and the second coverage run. The F_1 score between the two classification results returned high values for all three experiments (0.955, 0.968 and 0.976 respectively), which supports the assumption of temporally stable conditions for all three experiments.

For comparing the performance of the adaptive LSE-DP algorithm and the coverage runs, we applied a slightly different metric: We looked at the variance of the points having a mean value close to the threshold value. Thus, we evaluated the GP posterior mean over a dense set $\mathcal{S}_d$ of 10'000 points. From this, we extracted the set of points which are close to the threshold: $\mathcal{S}_p = \{x \in \mathcal{S}_d \,|\, h - h_{\text{eval}} < \mu(x) < h + h_{\text{eval}}\}$ with $h_{\text{eval}} = 1$. On these points, we then evaluated the GP predictive variance. This gives a more detailed evaluation of the regions around the threshold value, which we want to focus on.

Figure 8 shows the median (bold) and the 10% and 90% quantiles of the variance of $\mathcal{S}_p$ evaluated for both methods after each full pass over the transect. The uniform coverage runs outperform LSE-DP in the beginning as they cover the space more efficiently. The adaptive planner however is able to focus on the regions of interest more closely towards the end of the runs. These results support two conclusions. First, the uniform coverage runs decrease the uncertainty efficiently over the entire field. Second, Figure 8 shows that the reduction in variance decreases, emphasizing the fact that the uniform distribution of samples along the grid is not focusing

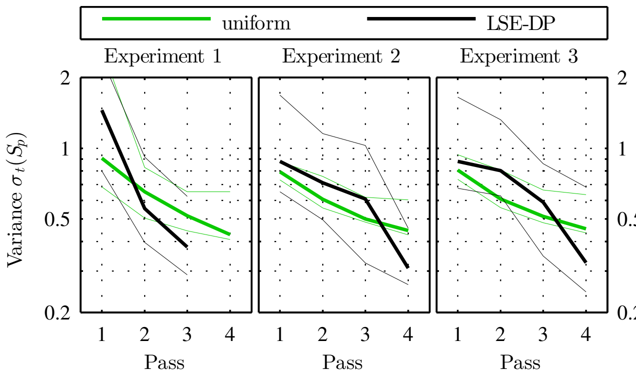

Fig. 8. The graph shows the median of the predicted variance around the threshold and the thin lines the 10% and 90% quantiles, respectively. Note: logarithmic y-axis.

on the regions of interest. Only by specifically and adaptively focusing on sampling the regions of interest enables LSE-DP to decrease the variance further. Finally, LSE-DP decreases the uncertainty around the boundary to 73%, 68% and 71% for the three experiments when compared to the results of the uniform coverage runs.

VI. Conclusion

In this paper, we addressed the problem of adaptively estimating level set boundaries in a scalar field using a mobile sensing agent. For this, we presented the LSE-DP algorithm, which implements a receding horizon planning scheme in order to plan non-myopically and overcome the well-known problems of greedy path planners. Furthermore, we applied our approach to a real environmental application, which consists of the monitoring of a toxic cyanobacteria in a lake using a robotic boat. We implemented the algorithm on the robot and demonstrated the fully autonomous execution of field experiments on the lake, during which the planning software ran fully integrated on the boat. The results from three field experiments over a total distance of 18 km demonstrated that the LSE-DP planner successfully focuses exploration on the desired boundaries and achieves higher confidence levels than simple coverage methods in the same amount of time. In the context of the application presented, our results will allows biologists to rapidly focus their research efforts on a subpart of the lake with the boat being able to identify *in-situ* regions of interest or punctual events.

In the future we would like to generalize our approach to higher dimensions and along with that loosen the constraints on the planning graph. Furthermore it would be interesting to incorporate temporal dynamics and conduct experiments over longer time periods.

ACKNOWLEDGMENT

This work was funded by the Swiss National Science Fund (grant number CR2212-130023). The authors would like to thank Hannes Sommer and Matthias Mock for their support during many hours of testing on the lake.

References

[1] M.-È. Garneau, T. Posch, G. Hitz, F. Pomerleau, C. Pradalier, R. Siegwart, and J. Pernthaler, "Short-term displacement of Planktothrix rubescens (cyanobacteria) in a pre-alpine lake observed using an autonomous sampling platform," *Limnology and Oceanography*, vol. 58, no. 5, pp. 1892–1906, 2013.

[2] T. Posch, O. Köster, M. M. Salcher, and J. Pernthaler, "Harmful filamentous cyanobacteria favoured by reduced water turnover with lake warming," *Nature Climate Change*, vol. 2, pp. 809–813, July 2012.

[3] G. Hitz, F. Pomerleau, M.-È. Garneau, C. Pradalier, T. Posch, J. Pernthaler, and R. Y. Siegwart, "Autonomous Inland Water Monitoring: Design and Application of a Surface Vessel," *Robotics Automation Magazine, IEEE*, vol. 19, no. March, pp. 62–72, 2012.

[4] M. Dunbabin and L. Marques, "Robots for Environmental Monitoring: Significant Advancements and Applications," *Robotics & Automation Magazine, IEEE*, vol. 19, no. 1, pp. 24–39, 2012.

[5] J. Gould, D. Roemmich, S. Wijffels, H. Freeland, M. Ignaszewsky, X. Jianping, S. Pouliquen, Y. Desaubies, U. Send, K. Radhakrishnan, K. Takeuchi, K. Kim, M. Danchenkov, P. Sutton, B. King, B. Owens, and S. Riser, "Argo Profiling Floats Bring New Era of In Situ Ocean Observations," *Eos Trans. AGU*, vol. 85, no. 19, pp. 190–191, 2004.

[6] A. Krause, A. Singh, and C. Guestrin, "Near-optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies," *The Journal of Machine Learning Research*, vol. 9, pp. 235–284, 2008.

[7] J. Binney, A. Krause, and G. Sukhatme, "Informative Path Planning for an Autonomous Underwater Vehicle," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 4791–4796.

[8] R. N. Smith, M. Schwager, S. L. Smith, B. H. Jones, D. Rus, and G. S. Sukhatme, "Persistent Ocean Monitoring with Underwater Gliders: Adapting Sampling Resolution," *Journal of Field Robotics*, vol. 28, no. 5, pp. 714–741, 2011.

[9] G. Hollinger and S. Singh, "Proofs and experiments in scalable, near-optimal search by multiple robots," in *Proceedings of Robotics: Science and Systems IV, Zurich, Switzerland*, 2008.

[10] K. H. Low, J. Chen, J. M. Dolan, S. Chien, and D. R. Thompson, "Decentralized Active Robotic Exploration and Mapping for Probabilistic Field Classification in Environmental Sensing," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 2012, pp. 105–112.

[11] G. A. Hollinger, B. Englot, F. S. Hover, U. Mitra, and G. S. Sukhatme, "Active Planning for Underwater Inspection and the Benefit of Adaptivity," *The International Journal of Robotics Research*, vol. 32, no. 1, pp. 3–18, Nov. 2012.

[12] A. Singh, R. Nowak, and P. Ramanathan, "Active Learning for Adaptive Mobile Sensing Networks," in *2006 5th International Conference on Information Processing in Sensor Networks*, 2006, pp. 60–68.

[13] A. Gotovos, N. Casati, G. Hitz, and A. Krause, "Active Learning for Level Set Estimation," in *International Joint Conference on Artificial Intelligence*, 2013.

[14] M. Batalin, M. Rahimi, Y. Yu, and D. Liu, "Call and Response: Experiments in Sampling the Environment," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, 2004, pp. 25–38.

[15] A. Singh, A. Krause, and W. J. Kaiser, "Nonmyopic Adaptive Informative Path Planning for Multiple Robots," *Center for Embedded Networked Sensing (CENS) Technical Report*, vol. 70, pp. 1843–1850, 2009.

[16] N. A. C. Cressie, *Statistics for Spatial Data*. Wiley, 1991.

[17] C. K. I. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, USA, 2006.

[18] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.