Path Efficient Level Set Estimation for Mobile Sensors

Lorenzo Bottarelli
Computer Science Department
University of Verona
lorenzo.bottarelli@univr.it

Jason Blum
Robotic Institute
Carnagie Mellon University and Platypus LLC
jasonblu@andrew.cmu.edu

Manuele Bicego
Computer Science Department
University of Verona
manuele.bicego@univr.it

Alessandro Farinelli
Computer Science Department
University of Verona
alessandro.farinelli@univr.it

ABSTRACT

The interest in using robotic sensors for monitoring spatial phenomena is steadily increasing. In the context of environmental analysis, operators typically focus their attention where measurements belong to a region of interest (e.g., when monitoring a body of water we might want to determine where the pH level is above a critical threshold). Most of the previous work in the literature represents the environmental phenomena with a Gaussian Process model, and then uses such a model to determine the best locations for measurements [3, 7]. In this paper we consider a specific scenario where a mobile platform with low computational power can continuously acquire measurements with a negligible cost. In this scenario, we seek to reduce the distance traveled by the mobile platform as it gathers information and to reduce the computation required by this path selection process. Starting from the LSE algorithm [7], we propose two novel approaches, PULSE and PULSE-batch, that exploit a new fast path selection procedure. We evaluate the effectiveness of our approaches on two datasets: a dataset of the pH level of the water, acquired with a mobile watercraft, and a publicly available dataset that represents CO_2 maps. Results show that our techniques can compute informative paths with a computation time that is an order of magnitude lower than other techniques.

CCS Concepts

•**Computing methodologies** → **Motion path planning; Intelligent agents; Mobile agents;**

Keywords

Informative Path Planning; Level Set Estimation; Mobile Sensors

SAC 2017, April 03-07, 2017, Marrakech, Morocco

DOI: http://dx.doi.org/10.1145/3019612.3019707

1. INTRODUCTION

With environmental monitoring we identify the analysis and actions performed to characterize and monitor the quality of the environment. This encompasses the collection of the information from the environment and the generation of the model that represents the specific phenomena of interest. This requires the collection of large data sets, frequently in harsh conditions. In recent years the use of unmanned vehicles for monitoring spatial phenomena has gained increasing attention. One example of a environmental monitoring application could be represented by the analysis of waters in a lake. In this case the analysis focuses on the generation of a model that describes how crucial parameters such as the pH level vary across the surface. This monitoring operation could be performed through the use of autonomous surface vessels (ASVs) such as the one showed in Figure 1, or by a heterogeneous system composed of marine, terrestrial and airborne platforms [5].

Figure 1: Platypus Lutra equipped with pH, Dissolved oxygen, temperature and electrical conductivity sensors; the computational board is composed by an Arduino Due and a smartphone.

When deploying unmanned vehicles for environmental monitoring, the data collection process must consider limited resources such as time and energy that limit the operation range of the platforms. The objective is to acquire a sufficient amount of data to generate an accurate model of the environmental phenomena of interest. In this context [9], it is important to select an informative path for the mobile agents in order to acquire as much information as possible while reducing the total traveled distance and hence the time and energy required to perform the analysis. Moreover, autonomous mobile systems are usually equipped with low computational capacity; if the path selection procedure is

performed on board during the monitoring operation, lower computational complexity is highly desirable.

We can explore different path selection strategies [12]. Traditional nonadaptive (offline) methods generate the path before any observations are made. In contrast, adaptive (online) methods plan the path based on the previously collected data. These adaptive techniques [2, 10, 13] incrementally generate the model of the environmental phenomena of interest during the data collection phase and focus the information collection process on specific regions of the environment where the phenomena exhibits critical values.

For example, in a lake, such a region would encompass the locations where the water's pH or dissolved oxygen level is considered dangerous for the environment. In another example, we might want to detect contours of biological or chemical plumes. This problem is typically referred to as the "level set estimation problem" in the literature [8].

Previous work on the level set estimation problem such as the one proposed by Dantu and Sukhatme [4] focuses on a network composed by a combination of static and mobile sensors. In Gotovos et al. [7] the proposed LSE algorithm uses Gaussian Processes (GPs) to identify sampling points that reduce uncertainty around a given threshold level of the modeled function. The authors obtain a near-optimal classification for every location of the space with a low number of sampled locations. In their contribution the main algorithm does not explicitly take into account the path between the sampling location. The authors propose a *batch* variant where a set of new sampling locations is selected in a batch such that it is possible to compute an efficient path between these points.

Hitz et al. [8] describes a method designed for ASVs equipped with a probe that allows an aquatic sensor to be lowered into the water. Their LSE-DP algorithm, built on the LSE algorithm from Gotovos et. al [7], uses a dynamic programming approach with a receding horizon to plan a feasible sampling path for the probe within a predefined vertical transect plane.

More recently, Bottarelli et al. [3] proposed a new algorithm also built on the LSE algorithm, but specifically designed for mobile continuous sampling sensors where the cost to perform an individual measurement is negligible. In this setting the most crucial issue for the data collection process is the energy required by the sensor to move. Hence, in their work the authors aim at reducing the total traveled distance required by the agent to achieve near-optimal classification of the analyzed regions, rather than the number of samples extracted during the execution of the path (which is an important criteria for previous work). The proposed SBOLSE algorithm uses an orienteering formulation [6] for the level set estimation problem. However, even with the topological skeletonization technique presented by the authors, the approach requires a computational effort that limits its use with the on-board computation units of standard mobile platforms.

In this paper, we also address the level set estimation problem by using adaptive techniques to plan efficient paths for continuous-sampling mobile sensors. Our main objective is to determine an informative path with a fast procedure. Specifically our techniques are motivated by the hardware specification of the platform shown in Figure 1. This platform is equipped with various sensors that can measure pH, dissolved oxygen, temperature and electrical conductivity with sampling rates between 1 and 10 Hz. The computational hardware is composed of an Arduino Due board and an Android smartphone. In our setting the energy required to perform a measurement is negligible. The important issues we have to address are the battery lifetime and the time required for the selection of an informative path. The main contributions of this paper to the state of the art are:

- We propose a novel algorithm called PULSE (Path-Update Level Set Estimation) for selecting measurement paths. The algorithm is specifically designed for continuous-sampling mobile sensors and aims at reducing the computation time required to determine an efficient path while achieving near optimal classification.
- We propose a batch variant of the PULSE algorithm, PULSE-batch, that trades off computation time and total traveled distance.
- We empirically evaluate our algorithms on a real world dataset of water pH level and on synthetic datasets extracted from CO_2 maps. We show that our approaches are better in terms of computation time required to compute a short path, while achieving a near optimal classification when compared to the state of the art techniques for level set estimations.

2. PROBLEM STATEMENT AND BACKGROUND

Following [7] and [3], we formalize the level set estimation as an active learning problem, where we want to select a path for the mobile sensor so as to optimize the information gathering process.

An unknown scalar field represents the environmental phenomena of interest, and every location in space has an associated scalar value. More formally, given a set of locations $D \subseteq \mathbb{R}^d$ and a threshold value h, we want to model the unknown scalar field $f : \mathbb{R}^d \mapsto \mathbb{R}$ in order to classify all the locations $x \in D$ into either the superlevel set $H = \{x \mid f(x) > h\}$ or the sublevel set $L = \{x \mid f(x) \leq h\}$. We model the scalar field with a Gaussian Process (GP) [11]. The problem asks to select the set of locations x_i where to perform (noisy) measurements $y_i = f(x_i) + e_i$ while minimizing the total traveled distance required for the sensor to analyze these locations.

2.1 Gaussian Processes

Gaussian Processes (GPs) offer a way to model unknown functions without using parameters and are a widely used tool in machine learning [11]. In this work the unknown function to be modeled using a GP is the unknown scalar field f of the environmental phenomena of interest. A GP is defined by a mean function $\mu(x)$ (that can be assumed to be zero without loss of generality) and by a kernel function (covariance function) $k(x, x')$ which represents the smoothness properties of the modeled function. A GP can then be denoted as $\mathcal{GP}(\mu(x), k(x, x'))$.

We will consider a set of noisy measurements $Y_t = \{y_1, y_2, \cdots, y_t\}$ taken at locations $X_t = \{x_1, x_2, \cdots, x_t\}$ and assume that $y_i = f(x_i) + e_i$ where $e_i \sim \mathcal{N}(0, \sigma_n^2)$ (i.e., measurement noise with zero mean). Given the GP prior $\mathcal{GP}(0, k(x, x'))$, the posterior over f is still a GP and its mean and variance can be computed as follows [11]:

$$\mu_t(x) = \mathbf{k}_t(x)^T(\mathbf{K}_t + \sigma_n^2\mathbf{I})^{-1}Y_t \quad (1)$$

$$\sigma_t^2(x) = k(x,x) - \mathbf{k}_t(x)^T(\mathbf{K}_t + \sigma^2\mathbf{I})^{-1}\mathbf{k}_t(x) \quad (2)$$

where $\mathbf{k}_t(x) = [k(x_1,x), \cdots, k(x_t,x)]^T$
and $\mathbf{K}_t = [k(x,x')]_{x,x' \in X_t}$

Using these equations, we can build the GP with the new measurements acquired by the sensor, however the update of the posterior is computationally expensive as it requires inverting an $n \times n$ matrix where n is the number of the samples acquired. In real-world applications n can be thousands of elements. The batch variant of our algorithm seeks to reduce the frequency of this computation.

2.2 LSE

Given the area of interest, we can discretize it into a grid where each element represents a small portion of the surface. These elements compose the set of sample locations D, and we want to classify each location $x_i \in D$ into two sets H or L with respect to a threshold level h.

Following the LSE algorithm [7], we use the inferred mean (1) and variance (2) from the GP to construct an interval:

$$Q_t(x) = \left[\mu_{t-1}(x) \pm \beta_t^{1/2}\sigma_{t-1}(x)\right] \quad (3)$$

for any $x \in D$. The parameter β represents a scaling factor for the interval.

In order to classify every point x into H or L, they define the following confidence interval using the the intersection of all previous $Q_t(x)$ intervals for every point x:

$$C_t(x) = \bigcap_{i=1}^{t} Q_i(x) \quad (4)$$

The classification of a point x depends on the position of its confidence interval with respect to the threshold level h. Specifically, for each location $x \in D$ if its confidence interval $C_t(x)$ lies entirely above h, then $f(x) > h$ with high probability, and we can classify x into the superlevel set H. Similarly, when the entire $C_t(x)$ lies below h then we can classify x into the sublevel set L. These conditions are relaxed with an accuracy parameter ϵ as shown in the following equations:

$$H_t = \{x \mid min(C_t(x)) + \epsilon > h\} \quad (5)$$

$$L_t = \{x \mid max(C_t(x)) - \epsilon \leq h\} \quad (6)$$

At time t, for every point with a confidence interval that crosses the threshold, we have to defer the decision until more information is available. We identify the set of unclassified locations:

$$U_t = D \setminus (L_t \cup H_t) \quad (7)$$

In order to classify the points in U_t according to the equations (5) and (6) we have to acquire more data selecting new sampling locations $x_i \in U_t$. To this end, the algorithm at each iteration uses the confidence interval for each unclassified point to derive the following ambiguity value:

$$a_t(x) = min\{max(C_t(x)) - h, h - min(C_t(x))\} \quad (8)$$

The point x_t with the highest ambiguity value represents the most interesting location. As such, it becomes the next point to measure.

In addition to the LSE algorithm, Gotovos et al. [7] discuss the batch version where multiple locations are selected by taking mutual information into account. Although the main goal of their approach is to select multiple locations and to compute an efficient path between them, in both cases their assumption is that the process of acquiring a new point of data is costly. Therefore their main goal is to minimize the number of sampling locations. Moreover, during the movement of the mobile agent from one location to next, the process does not acquire any further data.

3. PULSE ALGORITHM

Using the LSE technique, the mobile sensor is guided toward the most informative points. LSE assumes that the mobile sensor moves from the current position to the next selected location following a straight line.

In what follows we present our Path-Update LSE (PULSE) algorithm inspired by LSE, but specifically designed for continuous sampling sensors for which i) the cost required to perform an individual measurement is negligible, ii) it is necessary to optimize the total path of the agent in order to reduce the battery consumption and iii) we need an efficient path selection procedure. The proposed technique determines an informative path in order to reach the most interesting location (i.e. the point in space with the highest ambiguity about its classification) moving from the current position through points that still have to be classified. In this way it increases the information to path length ratio.

Algorithm 1 PULSE algorithm

Input: set D, threshold h, accuracy parameter ϵ, prior known data $X \subset D$, starting location x_{start}
Output: sets H and L

1: $t \leftarrow 0$
2: $x_0 \leftarrow x_{start}$
3: $H_0 \leftarrow \emptyset, L_0 \leftarrow \emptyset, U_0 \leftarrow D$
4: **while** $H_t \cup L_t \neq D$ **do**
5: $\quad t \leftarrow t + 1$
6: $\quad$ Compute GP posterior $\mu(x)$ and $\sigma^2(x)$ for all $x \in U_t$
7: $\quad$ Classify and update H_t, L_t, U_t according to LSE[7]
8: $\quad x_{t-1} \leftarrow x_t$
9: $\quad x_t \leftarrow$next location according to LSE [7]
10: $\quad path \leftarrow$ pathSelection(x_{t-1}, x_t, U)
11: $\quad$ Execute $path$
12: $H \leftarrow H_t, L \leftarrow L_t$

The pseudo-code of Algorithm 1 describes the steps of our PULSE approach. The algorithm maintains three sets of points: the current superlevel H_t and sublevel L_t sets, as well as the set of unclassified points U_t. At each iteration t we update the Gaussian Process posterior by integrating the new information gathered at the preceding iteration (line 6). Then we compute the confidence intervals $C_t(x)$ for each point $x \in U_t(x)$, classify them in one of the three sets and then compute the next sample to be evaluated using the ambiguity defined by equation (8) (line 7). We then compute a path between the current location x_{t-1} and the selected point x_t using the path selection procedure (Algorithm 2) The algorithm terminates when $H_t \cup L_t = D$, i.e. when all points are classified and thus $U_t = \emptyset$. Note that during the execution of the path (Algorithm 1, line 11) if an agent moves through locations that are already classified, these

Algorithm 2 pathSelection procedure

Input: last position x_{t-1}, next location x_t, unclassified elements U_t
Output: *path*

1: $i \leftarrow 0$
2: $x_{next_0} \leftarrow x_{t-1}$
3: $path \leftarrow x_{t-1}$
4: **while** $x_{next_i} \neq x_t$ **do**
5: $\quad i \leftarrow i + 1$
6: $\quad d \leftarrow dist(x_{next_{i-1}}, x_t)$
7: $\quad A \leftarrow \emptyset$
8: $\quad$ **for all** $x \in U_t$ **do**
9: $\quad\quad$ **if** $dist(x, x_t) < d$ **then**
10: $\quad\quad\quad A \leftarrow A \cup x$
11: $\quad x_{next_i} \leftarrow min_{x \in A}\Big(dist(x_i, x_{next_{i-1}})\Big)$
12: $\quad path \leftarrow path \cup x_{next_i}$

are re-evaluated and re-classified considering new acquired data. Moreover, note that Algorithm 1 differs from the LSE Algorithm proposed in [7] only in the path selection procedure we employ. Hence, the convergence property for LSE proved in [7] also holds in our case (see the discussion in Section 3.2 for more details).

3.1 Path Selection

At each time step t the algorithm keeps track of the starting position x_{t-1} of the platform (i.e. the last position) and the destination point assigned by the sample selection criteria, i.e. the most interesting point x_t. In order to select an informative path towards the destination the path selection procedure analyzes each point $x \in U_t$, i.e. locations that still have to be classified – therefore potentially carrying some useful information –, selecting a path $\{x_{t-1} = x_{next_0}, x_{next_1}, \cdots, x_{next_n} = x_t\}$ with $n \geq 1$. Note that the number of points touched by the agent, n, is automatically determined by the procedure. In the case of $n = 1$ the path corresponds to the straight line from the current position to the selected destination.

Each x_{next_i} point determined by the procedure meets the condition to always approach the destination point, i.e.

$$dist(x_{next_i}, x_t) < dist(x_{next_{i-1}}, x_t) \tag{9}$$

where $dist(x', x'')$ is the euclidean distance between locations x' and x''. In more detail, given the two points $x_{next_{i-1}}$ and x_t, the region of the space which contains points meeting this condition defines a convex area (see example in figure 2) and we call this area A_{t_i} (Algorithm 2, lines 8-10). The procedure analyzes all points $x_i \in U_t \cap A_{t_i}$ and selects as x_{next_i} the closest point from the previous location along the path (Algorithm 2, line 11).

3.2 Convergence analysis

Our path selection procedure selects only points that meet the condition in equation (9) (Algorithm 2, lines 6 and 9). As we build the path from x_{t-1} to x_t the area A_{t_i} shrinks and converges towards the destination point x_t. This allows the path to include informative points that lie inside this area (example in Figure 2), while ensuring that the path is going towards the most interesting point defined by the ambiguity measure $a_t(x)$ introduced in [7].

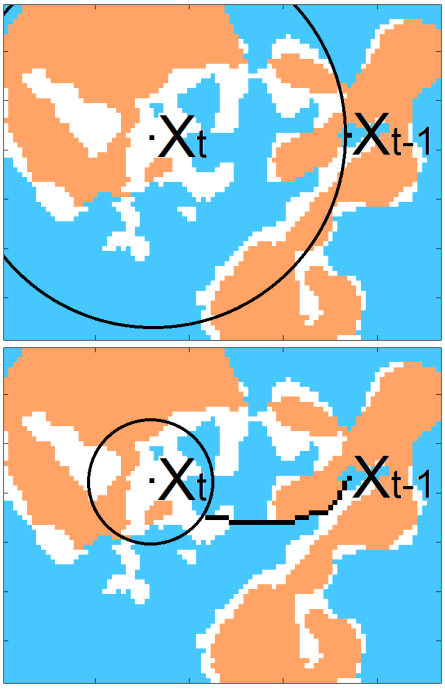


Figure 2: Example of runtime execution of the path selection procedure. The white areas represent location that are still unclassified. The circle represents the area A. On top the beginning of the procedure and on bottom we can observe the path that has been built after some iterations.

For what concerns the convergence analysis of our approach, notice that Gotovos et. al. with Theorem 1 in [7] proves the convergence of their LSE algorithm. Considering that PULSE uses the same classification and selection rules of LSE, the path selection procedure builds a path that terminates in the selected location, and that the information gathered along the path is never less than 0, the convergence is also valid for our algorithm.

3.3 Batch variant

We now propose a variant of the PULSE algorithm to select a set of informative locations in a single iteration (i.e. after a single Gaussian Process update). This will act as a trade-off between the computation time required and the path's efficiency. Following [7], we exploit the fact that the updated predictive variance in equation (2) depends only on the location of a measurement, not on the measurement value. Assuming we will obtain a new sample at some location, it is possible to evaluate the updated predictive variance, and thus the new ambiguity value, of every other point $x_i \in U_t$. This process is repeated adding the location with the new highest ambiguity to a set.

It is possible to compute an efficient path that visits all the locations in such a set. The order in which those locations should be visited is determined by solving a Travelling Salesman Problem (TSP) [1]. Once we have the order of locations to be visited, the path selection procedure (algorithm 2) is applied to all pairs of consecutive locations in order to obtain the final informative path. This algorithm allows us to trade off adaptivity in favor of a reduction of the total traveled distance required to classify all points $x \in D$.

4. EXPERIMENTS

In this section we now present the empirical evaluation of our proposed techniques comparing them with literature alternatives on the two datasets used in [3]. Specifically we compare our PULSE and PULSE-batch with two variants of the LSE and LSE batch algorithm from Gotovos et al. [7] and with the SBOLSE algorithm proposed by Bottarelli et al. [3]. The aim of this empirical evaluation is to assess the gain in terms of computation time required by our techniques with respect to the state of the art for the level set estimation problem. Moreover, we also assess the quality of the selected path, showing that our techniques are competitive in terms of total traveled distance required to obtain a near optimal classification. We identify the algorithms as follows:

- **PULSE:** Our algorithm as explained in section 3.
- **PULSE$_{bXX}$:** This algorithm is the batch variant of PULSE as described in section 3.3.
- **CS:** This is a variant of LSE as described in [7] to meet the continuous measuring setting. Locations on the straight line between the last position and the next selected point are analyzed, simulating a continuous sampling sensor.
- **CS$_{bXX}$:** Similar to CS, this is a variant of LSE batch as described in [7] to meet the continuous measuring setting.
- **SBOLSE:** This is the algorithm proposed by Bottarelli et al. in [3].

In the two batch versions, XX identifies the cardinality of the batch set, i.e. the number of locations in a TSP.

As previously done in [7] and [3], we assess the accuracy of the classification using the F_1-score. This is typically used in information retrieval to measure the accuracy of binary classification. Here we consider the locations in the superlevel set as positives and the locations in the sublevel set as negatives. All the described algorithms have been implemented and tested using MATLAB R2016a on a AMD FX 6300 processor with 16GB RAM.

4.1 Real data experiments

The real dataset consists of measurements of the pH level extracted from waters of the Persian Gulf near Doha, Qatar using the boat in Figure 1. The data forms a 68×93 grid where each element represents a sampling location x_i that must be classified with respect to a given threshold. Each point of the grid represents 0.5 square meters of the surface that has been analyzed. The value associated with that location is the average of all the samples extracted by the sensors while moving the boat in that portion of the surface.

In our experiments we applied three different thresholds (7.40, 7.42 and 7.44) to classify the scalar field. As done in previous approaches [3, 7], we performed tests to determine the β and ϵ parameter values that allow a high accuracy for all the algorithms. We then assessed the results starting from ten random initial priors composed by 10% of the points in the grid, for a total of 30 tests with every algorithm. These priors were used to fit the hyperparameters of an isometric Matérn-3 [11] covariance function. For the batch algorithms we performed tests with batches of different sizes. We did not observe a significant reduction of the total traveled distance with batches of size larger than 30. Thus, we carried out the comparisons with batches of 30 points.

Table 1: F_1-score, total traveled distance (meters) and computation time (seconds) using the real world pH dataset. $\overline{x}$ is the average of all experiments and $SE_{\overline{x}}$ is the standard error of the mean.

	F_1-score		**Traveled dist.**		**Comp. time**	
	$\overline{x}$	$SE_{\overline{x}}$	$\overline{x}$	$SE_{\overline{x}}$	$\overline{x}$	$SE_{\overline{x}}$
PULSE	97.46	0.063	587.8	10.82	11.1	0.27
PULSE$_{b30}$	97.43	0.060	518.7	6.68	63.5	0.89
CS	98.22	0.039	1560.8	18.58	38.1	0.49
CS$_{b30}$	97.47	0.055	671.7	13.71	82.4	1.74
SBOLSE	97.23	0.066	473.6	6.20	1006.2	45.99

As shown in Table 1 the F_1-score is higher than 97% and comparable between all the algorithms. With respect to the total traveled distance, our PULSE and PULSE$_{b30}$ algorithms perform very well, with a traveled distance that is lower than all but SBOLSE technique [3]. PULSE had a computation time two orders of magnitude less than SBOLSE. Compared to the variants of the LSE technique, PULSE can obtain both a lower traveled distance and reduced computation time (see Figure 3 for a graphical representation of the different paths).

4.2 Synthetic CO2 dataset experiments

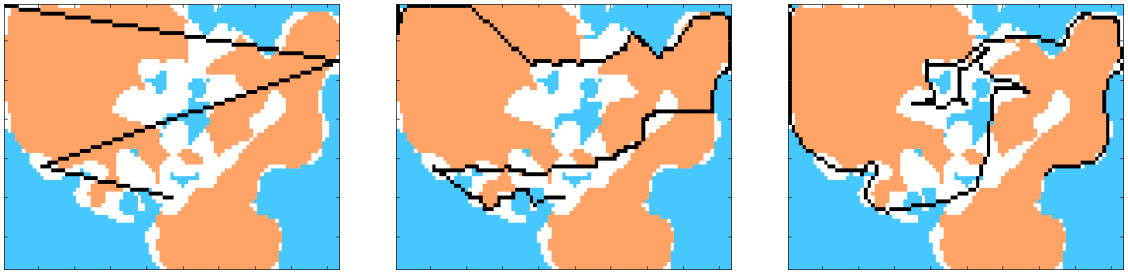
The synthetic dataset consists of ten 60×179 grids. The motivation for using this dataset is to test the techniques with more than 10,000 locations to classify. The dataset has been extracted from portions of CO_2 maps[1] in order to obtain a scalar field with a topology consistent with typical environmental phenomena. We assume that each location represents 1 square meter of surface to analyze, and we used a threshold value equal to 85% of the maximum value in the scalar field. As previously done with the real-world dataset, we determined a parameter setting that allowed a high accuracy with all the algorithms. We assessed the results with five random initial priors composed of 10% of the points in the grid. The priors were used to fit the hyperparameters of an isometric Matérn-3 [11] covariance function. With five priors per grid and ten grids, we performed a total of 50 tests with each algorithm.

Table 2: F_1-score, total traveled distance (meters) and computation time (seconds) using the synthetic CO_2 dataset, $\overline{x}$ is the average of all experiments and $SE_{\overline{x}}$ is the standard error of the mean.

	F_1-score		**Traveled dist.**		**Comp. time**	
	$\overline{x}$	$SE_{\overline{x}}$	$\overline{x}$	$SE_{\overline{x}}$	$\overline{x}$	$SE_{\overline{x}}$
PULSE	98.22	0.090	1709.4	35.37	23.9	0.75
PULSE$_{b30}$	98.23	0.092	1356.4	23.08	163.0	4.11
CS	98.66	0.071	5588.1	136.86	99.4	2.91
CS$_{b30}$	98.25	0.089	1782.7	34.05	223.5	5.08
SBOLSE	97.99	0.100	1355.6	26.16	3663.8	265.22

Result of these experiments are shown in Table 2. We observed similar results to real data experiments. The total

[1]http://oco.jpl.nasa.gov/galleries/gallerydataproducts/



(a) (b) (c)

Figure 3: Real dataset experiments. The white areas represent location that are still unclassified and black lines display the path selected by the algorithms: (a) CS, (b) PULSE and (c) SBOLSE.

traveled distances required by our PULSE and PULSE$_{b30}$ algorithms are lower than all but the SBOLSE technique. The PULSE$_{b30}$ traveled distance is less then one meter longer that of SBOLSE, but with a computation time of 163 seconds instead of 3663.8 seconds. The best algorithm in terms of computation time is PULSE with just 23.9 seconds of computation time required on average.

5. CONCLUSIONS

In this paper we proposed two new algorithms (PULSE and PULSE-batch) to solve the level set estimation problem considering an autonomous surface vessel equipped with continuous-measuring sensors and low computational capacity. Our techniques efficiently compute an informative path for the mobile agent in order to obtain a near-optimal classification. Results show that PULSE reduces computation time compared to the state of the art, while remaining competitive in distance traveled by the mobile sensor and classification accuracy. PULSE-batch represents a trade off between adaptivity and further reduction in total distance traveled.

6. ACKNOWLEDGMENTS

This work was supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 689341. This work reflects only the authors' view and the EASME is not responsible for any use that may be made of the information it contains.

7. REFERENCES

[1] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics)*. Princeton University Press, Princeton, NJ, USA, 2007.

[2] M. A. Batalin, M. Rahimi, Y. Yu, D. Liu, A. Kansal, G. S. Sukhatme, W. J. Kaiser, M. Hansen, G. J. Pottie, M. Srivastava, and D. Estrin. Call and response: Experiments in sampling the environment. In *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, pages 25–38, New York, NY, USA, 2004. ACM.

[3] L. Bottarelli, M. Bicego, J. Blum, and A. Farinelli. Skeleton-based orienteering for level set estimation. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 1256–1264, 2016.

[4] K. Dantu and G. Sukhatme. Detecting and tracking level sets of scalar fields using a robotic sensor network. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3665–3672, April 2007.

[5] M. Dunbabin and L. Marques. Robots for environmental monitoring: Significant advancements and applications. *Robotics Automation Magazine, IEEE*, 19(1):24–39, March 2012.

[6] B. L. Golden, L. Levy, and R. Vohra. The orienteering problem. *Naval Research Logistics (NRL)*, 34(3):307–318, 1987.

[7] A. Gotovos, N. Casati, G. Hitz, and A. Krause. Active learning for level set estimation. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 1344–1350. AAAI Press, 2013.

[8] G. Hitz, A. Gotovos, F. Pomerleau, M.-E. Garneau, C. Pradalier, A. Krause, and R. Siegwart. Fully autonomous focused exploration for robotic environmental monitoring. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2658–2664, May 2014.

[9] G. A. Hollinger and G. S. Sukhatme. Sampling-based robotic information gathering algorithms. *Int. J. Rob. Res.*, 33(9):1271–1287, Aug. 2014.

[10] M. Rahimi, R. Pon, W. J. Kaiser, G. S. Sukhatme, D. Estrin, and M. Srivastava. Adaptive sampling for environmental robotics. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 4, pages 3537–3544 Vol.4, April 2004.

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

[12] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser. Efficient informative sensing using multiple robots. *J. Artif. Int. Res.*, 34(1):707–755, Apr. 2009.

[13] A. Singh, R. Nowak, and P. Ramanathan. Active learning for adaptive mobile sensing networks. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, IPSN '06, pages 60–68, New York, NY, USA, 2006. ACM.