A Graph-Based Approach to Boundary Estimation With Mobile Sensors

Sean O. Stalley, Dingyu Wang, Gautam Dasarathy, *Senior Member, IEEE*, and John Lipor, *Member, IEEE*

***Abstract*—We consider the problem of adaptive sampling for boundary estimation, where the goal is to identify the two-dimensional spatial extent of a phenomenon of interest. Motivated by applications in estimating the spread of wildfires with a mobile sensor, we present a novel graph-based algorithm that is efficient in both the number of samples taken and the distance traveled. The key idea behind our approach is that by sampling locations close to known cut edges (edges whose vertices lie on opposite sides of the boundary), we can reliably find additional cut edges. Our approach repeats this process of using the newly discovered cut edges to find additional cut edges, eventually identifying all vertices lying adjacent to the boundary. We show that our method achieves both a sample complexity and a distance traveled that are within a constant factor of the optimal values. Moreover, the computational complexity of determining sample locations and paths is $O(1)$, making its deployment on mobile sensors highly realistic. Experimental results on both synthetic and historical wildfire data show that our proposed algorithm outperforms existing methods in terms of sample complexity, distance traveled, and computation time.**

***Index Terms*—Sensor-based control, machine learning for robot control, environment monitoring and management.**

I. Introduction

RAPIDLY sensing and estimating phenomena of interest is a fundamental problem to scientists and engineers, and the recent development of both low-cost robots and on-board sensors has enabled safe, persistent monitoring across a variety of applications. As a motivating example, we consider the problem of estimating wildfire boundaries using a sensor mounted to an unmanned aerial vehicle (UAV). In this setting, it is essential to estimate the boundary as quickly as possible, both to provide responders with the most accurate information and to account for nonstationarity in the wildfire front. Toward this aim, a key problem is that of developing algorithms that can reliably estimate the boundary while minimizing the total sampling time, a function of both the number of measurements taken and the distance traveled.

Recent research has shown that active learning methods can be applied to improve the performance of mobile sensing platforms when tasked with a variety of problems, such as localization and mapping with aerial robots [1]–[3], selecting the most informative images collected by a robot [4], improving movement predictability when sharing an environment with humans [5], intelligent mobile sensor placement for improved environmental model accuracy [6], [7], and improved positional accuracy of autonomous underwater vehicles [8]. Advances in active learning have resulted in algorithms that achieve near-optimal sample complexity [9], [10] as well as the ability to incorporate non-uniform label costs [11]–[13]. An important application of these methods has been one of discovering the spatial extent of some phenomenon using mobile sensors [6], [14]–[20]. Traditional active learning techniques are sample efficient but do not account for the cost associated with the distance traveled by the mobile sensor. Existing adaptations of active learning to this problem either rely on strong modeling assumptions [21], are limited to very restrictive cases [18], or treat this cost myopically [13].

Among approaches from active learning with low sample complexity, the shortest-shortest path (S^2) algorithm [9] is shown to exhibit a zig-zagging behavior that traces the class decision boundary in a manner that is likely to reduce the distance traveled. However, this behavior is not guaranteed, and hence S^2 may sample a sequence of points with arbitrarily long path length. Further, the sample selection process for S^2 requires computing all pairs of shortest paths between nodes in the graph, incurring a large computational cost that makes it impractical for deployment on mobile sensors.

In this work, we present a novel graph-based algorithm for active boundary estimation that overcomes the shortfalls of existing methods. The key idea is that we reduce this problem to one of level set estimation of a graph function; by sampling locations near known *cut edges* (graph edges whose vertices lie on opposite sides of the boundary), we can reliably find more cut edges and efficiently determine all vertices lying adjacent to the boundary. We show that our method is optimal in terms of sample complexity (for the graph reduction) and nearly optimal in terms of distance traveled. Further, experiments on synthetic and real wildfire boundaries show that our algorithm is computationally efficient in choosing sample locations, making its deployment on a mobile sensor realistic.

Manuscript received September 9, 2021; accepted January 10, 2022. Date of publication January 25, 2022; date of current version March 8, 2022. This letter was recommended for publication by Associate Editor P. Vasseur and Editor E. Marchand upon evaluation of the reviewers' comments. This work was supported in part by the NSF under Grants OAC-1934776, CNS-2003111, CCF-2007688, CIF-1850404, and CIF-2046175, and in part by the Office of Naval Research under Grant N00014-21-1-2615. *(Corresponding Author: Sean O. Stalley.)*

Sean O. Stalley and John Lipor are with the Department of Electrical & Computer Engineering, Portland State University, Portland, OR 97201 USA (e-mail: seanstalley@gmail.com; lipor@pdx.edu).

Dingyu Wang is with the Department of Computer Science & Engineering, University of Michigan, Ann Arbor 48109 USA (e-mail: wangdy@umich.edu).

Gautam Dasarathy is with the School of Electrical, Computer & Energy Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: gautamd@asu.edu).

Digital Object Identifier 10.1109/LRA.2022.3145977

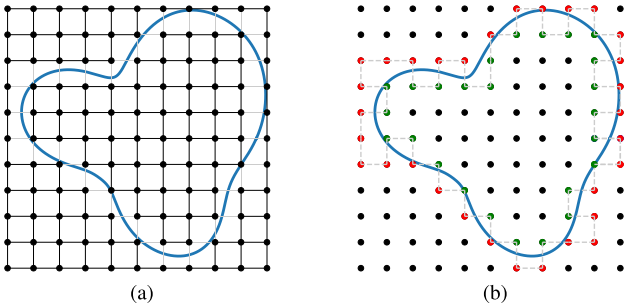

Fig. 1. (a) An example level set boundary ∂B (blue) and partition P_{10}. Gray lines indicate the cut edges in C that connect vertices on opposite side of the level set. (b) An example sampling pattern of the proposed CuP algorithm on ∂B. Green dots indicate measurements inside the boundary, red dots indicate measurements outside the boundary, and the gray dashed line depicts the path traveled during sampling. Black dots are unsampled locations.

II. Problem Formulation & Related Work

Given a function $f : [0,1]^2 \to \mathbb{R}$, the γ-level set of f is defined as

$$B = \left\{x \in [0,1]^2 : f(x) \geq \gamma\right\},$$

where γ is a critical value determined by the end user. In this work, we consider the case where the level set B consists of a single, simply-connected region. The problem of level set estimation can be viewed as one of estimating the boundary, denoted as $\partial B \triangleq \{x \in [0,1]^2 : f(x) = \gamma\}$, between the γ-level set B and its complement. Toward this end, we discretize and consider a set of potential sampling locations determined by a regular partition of $[0,1]^2$ into squares of side length $1/w$, denoted by P_w. Notice that w determines the accuracy of our estimate of the boundary. An example boundary and partition P_{10} are shown in Fig. 1(a).

Given a partition P_w, consider the undirected graph $G_w = (V, E)$ with vertices $V \subset [0,1]^2$ corresponding to the corners of each cell and edges E such that the resulting nodes are connected to their four nearest neighbors (see Fig. 1(a)). Let $g : V \to \{0,1\}$ define the labeling of V according to f, i.e., $g(v) = 1$ if $f(v) \geq \gamma$ and $g(v) = 0$ otherwise. Define the *cut set* with respect to g by $C = \{e_{x,y} \in E : g(x) \neq g(y)\}$, where $e_{x,y}$ denotes the edge connecting vertices x and y, and let ∂C denote the boundary of the cut set $\partial C = \{x \in V : \exists e \in C \text{ with } x \in e\}$. The boundary ∂C corresponds to all points in P_w whose corresponding cells intersect the boundary ∂B, and hence by uncovering ∂C we obtain an estimate of ∂B, where the approximation error decreases as w grows.

Since the sampling procedure will be performed by a mobile sampling vehicle, we wish to minimize not only the number of samples taken, but also the distance traveled throughout the sampling procedure. Hence our goal is to uncover the set ∂C while traveling a distance on par with the total length of the boundary ∂B.

In this work we assume that each sample yields the exact value $g(v)$, noting that noisy labels can be accommodated through repeated measurements as proven in Proposition 1 of [9]. Additionally, when comparing a fixed γ-level set against measurements $f(v)$ with Gaussian or Sub-Gaussian noise, Hoeffding's inequality shows us that the uncertainty of the resulting label $g(v)$ decreases exponentially with the number of repeated measurements.

A. Related Work

As stated above, a number of approaches exist to perform spatial sampling through the framework of active learning. Greedy approaches, such as those based on adaptive submodularity [12] have the flexibility to incorporate nonuniform costs and have strong theoretical guarantees. However, submodularity is fundamentally a property of set functions, and hence costs such as the distance between sequential samples cannot be incorporated. A nonmyopic graph-based method for adaptive sampling is introduced by [22] that exploits this submodularity to achieve near-optimal sampling paths when the environment is predictable. Unfortunately, this method was designed to solve the information path planning problem, and is not applicable to our problem of boundary estimation. A notion of submodular optimization with sequential dependencies was presented in the recent work [23], but the proposed algorithm relies on a reordering procedure that is not applicable to our problem.

Numerous approaches to adaptive sampling assume the underlying function f is a Gaussian process (GP), beginning with [16], [24], where confidence-based algorithms are presented for function optimization and level set estimation, respectively. These proceed by successively sampling points based on an upper/lower confidence bound related to the variance of the GP estimate. This approach is extended in [13], [17] and deployed on an autonomous surface vessel for spatial sampling. Another approach to adaptive sampling is given in [6] where the authors modify the approach from [24] to move mobile sensors throughout the sample space to improve accuracy of the function estimate. A similar method is used in [7] to uncover the global maximum of an unknown field using multiple mobile sensors. However, these approaches require the selection of an appropriate kernel, as well as a number of hyperparameters, both of which require existing data to fit. Moreover, these methods incur a computational cost that is cubic in the number of measurement locations considered, which may prohibit their use in real-time settings.

Distance-penalized active learning in one dimension was first considered in [18] and extended in [25], though neither approach provides optimality guarantees. In the recent work [19], the authors show that the one-dimensional distance-penalized boundary detection problem can be formulated as a stochastic shortest path problem, for which an optimal policy can be obtained using dynamic programming. In this case, a two-dimensional boundary can be estimated using a series of transects. However, determining the optimal number of transects and their impact on the overall cost remains an open problem.

In [26], the authors present a means of estimating a one-dimensional boundary of interest using a zig-zag pattern. A similar method capable of estimating a two-dimensional boundary is presented in [27], [28]. In this approach, the mobile sensor follows circular paths across the boundary, adaptively updating

the mobile sensor's trajectory whenever the boundary is crossed. Further improvements to the method of [27], [28] are made in [29] by constantly adapting the sensor's trajectory based on the current measurement. This allows the mobile sensor to more closely track the boundary, yielding a more accurate estimate. However, this method uses a gradient estimation technique which requires continuous-valued sensor readings and cannot be implemented from binary labels alone. Additionally, none of these approaches provide any theoretical guarantees, and all assume samples can be obtained continuously with negligible cost. Our empirical results (see Section IV) demonstrate that our proposed method requires fewer samples and travels a lower distance than the methods of [27], [28].

Most related to the work presented here, [9] presents S^2, a graph-based approach to active learning that exhibits this zig-zagging behavior, yielding strong theoretical guarantees under broad assumptions. The algorithm achieves near-optimal sample complexity for non-parametric learning by successively sampling the *shortest shortest path* between vertices lying in ∂C. While this algorithm is sample efficient, it requires performing the computationally-expensive task of finding the shortest shortest paths between all previously-unsampled vertices, making its deployment on a mobile sensor impractical. Further, in the case where there are multiple shortest paths, the algorithm breaks ties arbitrarily, resulting in unnecessarily large distances traveled between subsequent sample locations, making it a poor choice for distance-penalized applications.

III. Algorithm Description & Analysis

In this section, we describe the proposed algorithm for boundary estimation in two dimensions, which we refer to as *Cut Pursuit* (CuP). We show that CuP overcomes the shortcomings of S^2, uncovering the same cut edges while traveling a shorter distance and requiring much less computation between samples.

The CuP algorithm proceeds as follows. Assume without loss of generality that sampling begins at the point x_1 with $g(x_1) = 1$ and proceeds to the point x_2 with $g(x_2) = 0$ having distance $1/w$ from x_1. These points could be obtained either from prior knowledge, or from applying a one-dimensional spatial sampling algorithm such as that of [19]. In the notation above, the line segment connecting these two points is a cut edge lying in the set C, and the vertices x_1, x_2 correspond to adjacent corners of a single cell. In the general case where x_1 and x_2 are not adjacent, CuP begins by sampling along a path between x_1 and x_2 until an adjacent pair of locations with different values is found. We assume that the level set B is either fully contained within the sampling region (as in Fig. 1(a)) or that the initial locations x_1, x_2 are on one extreme of the domain.

By convention, we maintain our "current" cut edge by denoting the "inside" vertex a to be such that $g(a) = 1$, and the "outside" vertex b such that $g(b) = 0$, yielding the initial cut edge $e_{a,b} = e_{1,2}$. This edge is then removed from the graph ($G \leftarrow G \setminus e_{a,b}$) and the shortest path p between a and b is calculated. Our proposed algorithm then samples iteratively along this path, updating a and b when appropriate until another cut edge is found. Subsequent cut edges are then removed from the graph, at which point p is recalculated based on the "current" vertices a and b. The process of removing edges and finding paths repeats until no path exists between a and b, meaning the two vertices are elements of two separate graph components, and an estimate of the boundary between these two components has been found.

Algorithm 1: Cut Pursuit (CuP) for Boundary Estimation.

Input: Graph G, initial vertices x_1, x_2 where $g(x_1) = 1$ and $g(x_2) = 0$
Initialize: $a = x_1, b = x_2, p = \text{path}(a \leftarrow b)$
while p exists **do**
  while a and b are not adjacent **do**
    Obtain closest sample x_n in path p (unless previously sampled)
    if $g(x_n) = 0$ **then**
      $b \leftarrow x_n$
    else
      $a \leftarrow x_n$
    end if
  end while
  $G \leftarrow G \setminus e_{a,b}$
  $p = \text{path}(a \leftarrow b)$
end while

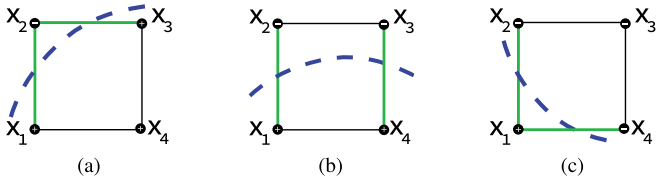

Fig. 2. Example sampling procedure of proposed CuP algorithm. The boundary is the blue dashed line and cut edges are depicted in green. The initial cut edge is $e_{a,b} = e_{1,2}$. Sampling vertex x_3 may reveal a cut edge $e_{3,2}$ (a). If x_3 does not reveal a cut edge, then sampling vertex x_4 will reveal a cut edge at either $e_{4,3}$ (b) or $e_{1,4}$ (c).

Let us consider (without loss of generality) the first shortest path p around a single cell (see Fig. 2). Denote the first vertex sampled along the first path p as x_3. If $g(x_3) = 1$, then $a = x_3$ and $e_{a,b} = e_{3,2}$ is a new cut edge. The edge $e_{a,b}$ is then removed from the graph ($G \leftarrow G \setminus e_{a,b}$) and p is recalculated (in $O(1)$ time, as described below). Otherwise, $b = x_3$ and no cut edge is discovered. In this case the next sample x_4 is the fourth vertex in the current cell. If $g(x_4) = 1$, then $a = x_4$ and $e_{a,b} = e_{4,3}$ is a new cut edge, and if $g(x_4) = 0$, then $b = x_4$ and $e_{a,b} = e_{1,4}$ is a new cut edge. In all cases a cut edge is discovered, and the next vertex to sample lies along the path p.

The pseudocode for this procedure is given in Alg. 1. After the initial cut edge is discovered and removed, the path p is always guaranteed to be the path around a single cell. This ensures that p always has exactly four nodes: a, b, and the two intermediate nodes in the cell. The intermediate nodes may have been members of a previous path p and therefore may have already been sampled. In this case the sample value is already known and the mobile sensor does not need to travel to this location again. Because the length of p is fixed to a constant,

finding the next path p (line 13 of Algorithm 1) has complexity $O(1)$. This also ensures that the inner loop (lines 4 through 11) is run at most two times, making these lines $O(1)$ as well. Therefore, every operation inside the outer loop (lines 3 through 14) of Algorithm 1 can be completed in $O(1)$, making the computational complexity of the sample selection/path planning procedure $O(1)$.

Fig. 1(b) shows an example of the sampling pattern selected by CuP, where the red dots correspond to locations x_n with $g(x_n) = 0$, the green dots correspond to $g(x_n) = 1$, and the gray dashed line denotes the path traversed throughout the sampling procedure. The figure shows that CuP achieves the desired zig-zagging behavior and does not sample any vertices away from the cut set.

A. Theoretical Results

In this section, we analyze the performance of the proposed CuP algorithm. We show that, under a mild assumption on the level set, CuP recovers the cut set exactly while maintaining a sample complexity and distance traveled within a constant factor of the optimal values.

Assumption 1: The level set B and inverse partition side length $w \in \mathbb{N}$ are such that the graph $G_w \backslash C$ consists of exactly two components.

Our main assumption implies that (1) the continuous-domain level set consists of a single, simply connected component, and (2) that the partition P_w is fine enough to maintain the connectedness of this set. This corresponds to boundaries that are guaranteed to be completely uncovered by the aggressive search phase of S^2, and for applications of interest such as air and water quality monitoring this assumption is likely to be satisfied for sufficiently large w [30], [31]. We remark that multiple connected components can be easily accommodated through either random search, following [9], or through prior knowledge indicating one point within each component. Level sets whose boundary is a Lipschitz function in one coordinate can be shown to satisfy this assumption, though these may be restrictive in practice. In the following theorem, we consider a more realistic setting, showing that level sets with radial Lipschitz boundaries satisfy Assumption 1.

Theorem 1: Assume that the level set boundary ∂B is a 2π-periodic function $r(\phi)$ that is bounded below by r_{min}, i.e., $r(\phi) \geq r_{min}$, $\forall \phi \in [0, 2\pi]$. Further, assume that r is K-Lipschitz with constant

$$K \leq \frac{\sqrt{r_{\min}^2 - \frac{1}{2w^2}} + \frac{1}{\sqrt{2}w} - r_{\min}}{\sin^{-1}\left(\left(\sqrt{2}wr_{\min}\right)^{-1}\right)}, \tag{1}$$

Then the corresponding graph G_w satisfies Assumption 1.

Proof: The theorem holds as long as no cell contains more than two cut edges. We consider a cell in the worst-case location and derive the Lipschitz constant for which this cell can have no more than two cut edges.

Consider a cell with vertices at locations $x_1, \ldots, x_4$ where vertices x_1 and x_3 both lie on the circle of radius $r_{\min}$ and are outside of the boundary. These two vertices, being as close as possible to the center of the circle, represent the largest possible difference in ϕ that two samples in the same cell can have while lying outside of the boundary. Fig. 3 shows the relevant geometry for such a cell. We denote the ϕ values of these two vertices ϕ_1 and ϕ_3, where $r(\phi_1) = r(\phi_3) = r_{\min}$. The orientation and location of this cell minimizes the maximum difference in r between these vertices and the farthest vertex from the center of the circle, denoted x_2. In order for this cell to contain four cut edges, both x_2 and x_4 must be inside the boundary, while x_1 and x_3 remain outside of the boundary. A cell in this orientation requires the smallest possible change in r over the largest possible change in ϕ to produce a cell with four cut edges. Thus, if our Lipschitz constant is small enough to ensure that Assumption 1 holds for this cell, it must also hold for all other cells in the sampling domain.

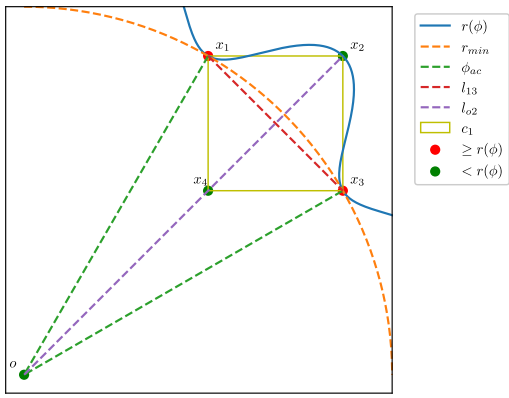

Fig. 3. Example case of a cell with two opposite vertices on the $r_{\min}$ circle and a boundary that creates four cut edges.

Now let us use the properties of this cell to derive a Lipschitz constant. The distance between the center o and the vertices x_1, x_3 is $r_{\min}$, and the distance between these two vertices is $\sqrt{2}/w$. We now have an isosceles triangle with known side lengths from which the angle between x_1 and x_3 can be derived:

$$\phi_1 - \phi_3 = 2\sin^{-1}\left(\left(\sqrt{2}r_{\min}w\right)^{-1}\right). \tag{2}$$

Due to the symmetry of the triangle and the cell in this orientation, $\phi_1 - \phi_2 = \phi_2 - \phi_3 = (\phi_1 - \phi_3)/2$.

Let us now compare $r(\phi_2)$, with $r(\phi_1)$ and $r(\phi_3)$, which are known to be $r_{\min}$. Note that $r(\phi_2)$ must be greater than the distance between o and x_2 for x_2 to lie inside the boundary. The distance from o to x_2 can be broken into two computable distances: the segment between x_2 and the intersection with line l_{13} between x_1 and x_3, and the segment between l_{13} and the origin o,

$$\begin{aligned}\operatorname{dist}(o, x_2) &= \operatorname{dist}(o, l_{13}) + \operatorname{dist}(l_{13}, x_2) \\ &= \sqrt{r_{\min}^2 - \tfrac{1}{2w^2}} + \tfrac{1}{\sqrt{2}w}.\end{aligned} \tag{3}$$

Therefore the change in r between ϕ_1 and ϕ_2 must be at least $\operatorname{dist}(o, x_2) - r_{\min}$ to produce such a labeling. The resulting inequality is derived from the change in ϕ given in (2) and the change in r given in (3). ■

Radial Lipschitz boundaries have been studied extensively in the context of non-parametric estimation [32] and are a subset of the box counting class [33]. Further, such boundaries are likely to occur in air quality monitoring contexts and align with the spatial models used to estimate particulate matter [34]–[36]. The requirement of a minimum radius on the boundary is due to the assumption that the boundary ∂B have a functional form and is not a strict requirement for our algorithm. A more precise characterization of level sets that satisfy Assumption 1 is a topic for our future research.

Next, we prove that CuP recovers the cut set boundary exactly and characterize its performance in terms of the number of samples required and distance traveled.

Theorem 2: Let B and w be such that Assumption 1 is satisfied. Then CuP uncovers the boundary estimate ∂C in at most $2|C|$ queries.

Proof: Consider all possible enumerations of labelings for a single cell. Note that of the possible enumerations, there are only cases with two or four cut edges. The cases with four cut edges occur when $g(x_1) = g(x_3) \neq g(x_2) = g(x_4)$, for $x_1, \ldots, x_4$ as in the proof of Theorem 1. Such a cell cannot exist unless there are more than two components in the graph; thus, if Assumption 1 is satisfied, all cells containing a cut edge must contain exactly two cut edges.

Now consider the boundary ∂B as an ordered set of boundary segments, with each segment being the portion of the boundary that is contained within an individual cell. Note that each boundary segment connects one cut edge to another subsequent cut edge, and by following along the entire boundary we will observe an ordered set of all the cut edges. The CuP algorithm iteratively discovers the cut edges following this order, and thus will discover all the cut edges.

Consider a single cell $c_k \in P_w$ with an initial cut edge corresponding to two vertices of c_k obtained from two previous queries. If Assumption 1 is satisfied, then there is exactly one other cut edge in c_k along the path p_k. As shown in Section III, CuP reveals a cut edge in every path p_k. and the maximum number of unsampled locations along every path p_k is two. Thus, the maximum number of queries needed to uncover all the cut edges is twice the number of cut edges. ■

The above sample complexity matches the aggressive search complexity of S^2 and is sharp, since a boundary ∂B consisting of a straight line will have $|\partial C| = 2|C|$. In such a case, CuP queries the minimum number of samples possible to recover ∂C exactly. On the other hand, the minimum value of $|\partial C|$ is $|C| + 1$, corresponding to the case where B contains only a single vertex of G_w. In this case, the sample complexity of CuP is within a constant factor of the optimum sample complexity.

Finally, we derive an upper bound on the distance traveled by the proposed algorithm in terms of the boundary length $\mathcal{L}(\partial B)$.

Theorem 3: Let B and w be such that Assumption 1 is satisfied, and assume ∂B has length at least $4/w$. CuP travels a distance of at most $4\mathcal{L}(\partial B)$ to uncover the boundary estimate, where $\mathcal{L}(\partial B)$ is the length of the boundary.

Proof: In the worst case, CuP visits two unsampled nodes in path p before identifying a new cut edge. To do this, it travels at most $2/w$. To uncover all the cut edges, it must travel $2|C|/w$. The resulting expression then follows from substituting in the bound in Lemma 1. ■

Lemma 1: The number of cut edges $|C|$ for a given boundary length $\mathcal{L}(\partial B)$ is at most $2w\mathcal{L}(\partial B)$ where w is the inverse side length, provided $\mathcal{L}(\partial B) \geq 4/w$.

Proof: Given a 2×2 array of cells within some graph G_w, consider the set of continuous boundary segments that intersect all four cells and intersect two outside edges of such an array. The length of such a boundary segment must be at least $2/w$, regardless of what outside edges the boundary intersects. Such a boundary segment can be considered optimal in the sense that it creates the largest possible number of cut edges over the smallest possible length of boundary. A boundary that is constructed of $k \geq 2$ such boundary segments therefore has a length of at least $2\,k/w$ and intersects $4\,k$ cut edges. ■

The maximum distance traveled by CuP is a constant factor of the boundary length, meaning the total distance traveled scales directly with the length of the boundary being estimated. This makes CuP near-optimal in terms of distance traveled. On the other hand, S^2 has no means of accounting for the distance traveled, and we will show empirically that breaking ties arbitrarily results in a distance that increases with w.

IV. Simulations

In this section, we compare the performance of our proposed algorithm with the S^2 algorithm of [9] as well as the adaptive and non-adaptive Bang-Bang algorithms proposed in [27], [28]. Specifically, we compare the performance of CuP to the "aggressive search phase" of S^2, described by lines 4 through 10 of Algorithm 1 in [9], as well as a mobile sensor following the fixed and adaptive Bang-Bang steering control policies described in Eq. (1) and (2) of [28], where the distance between measurements is set to the graph side length $1/w$ in order to compare to CuP and S^2. Both the fixed and adaptive Bang-Bang algorithms can be tuned to adjust the angle between subsequent samples. Additionally, the "adaptivity" of the adaptive Bang-Bang algorithm can be tuned. We tested both algorithms over a range of angles from 40^o to 140^o between samples and considered adaptive adjustments between 0% and 50% of the maximum angle per sample, reporting the best-performing results here.

We run the algorithms on synthetic boundaries as well as historical wildfire boundaries from the MTBS Burned Area Boundaries Dataset [37].[1]

We first characterize the performance of our proposed algorithm by drawing 100 random boundaries from a radial GP [38, Ch. 4] and check to ensure that each boundary satisfies Assumption 1 for inverse partition widths w ranging between 10 to 50. We then run the algorithms on each boundary for this range of w values and compare our results.

We also characterize the performance of the algorithm on real wildfire boundaries from the MTBS dataset. Specifically, we investigate the performance of these algorithms on wildfire boundaries in Oregon ranging in size from 1000 to 5000 acres by

[1]The source code for these experiments is available at `https://github.com/sstalley/CuP`

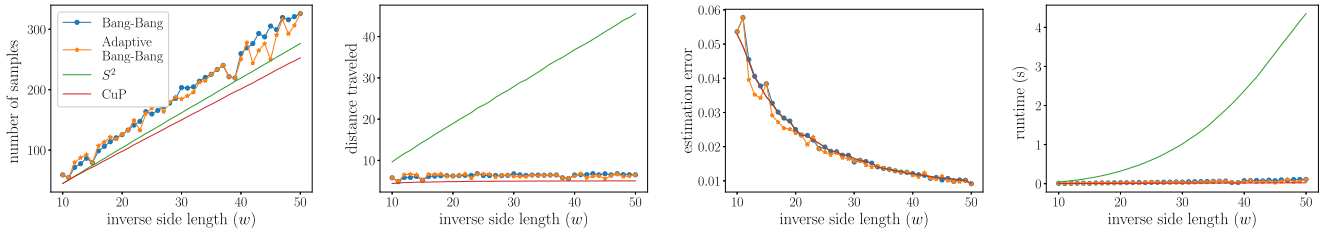

Fig. 4. Average performance of proposed CuP algorithm compared with Bang-Bang and S^2 algorithms on synthetic boundaries as a function of inverse side length w. (a) Number of samples required to uncover all cut edges. (b) Distance traveled. (c) Estimation error. (d) Computation time. For CuP the sample complexity is linear in w as predicted by Theorem 2, the distance traveled is independent of w, as predicted by Theorem 3, and the computation time is orders of magnitude lower than that of S^2 as predicted in Section III.

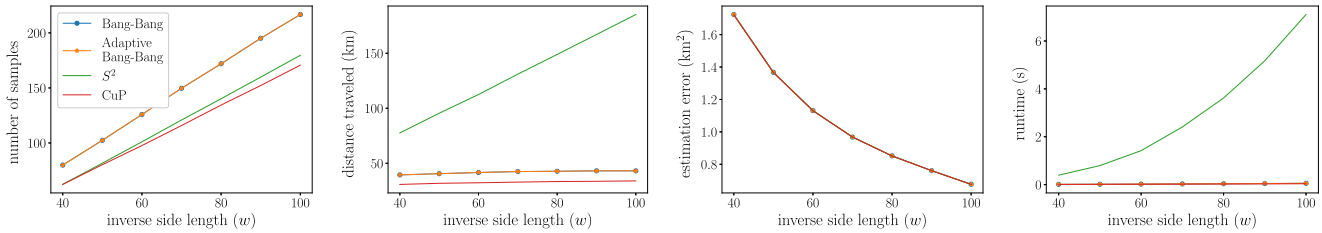

Fig. 5. Average performance of proposed CuP algorithm compared with Bang-Bang and S^2 algorithms on historical wildfire data. (a) Number of samples required to uncover all cut edges. (b) Distance traveled. (c) Estimation error. (d) Computation time.

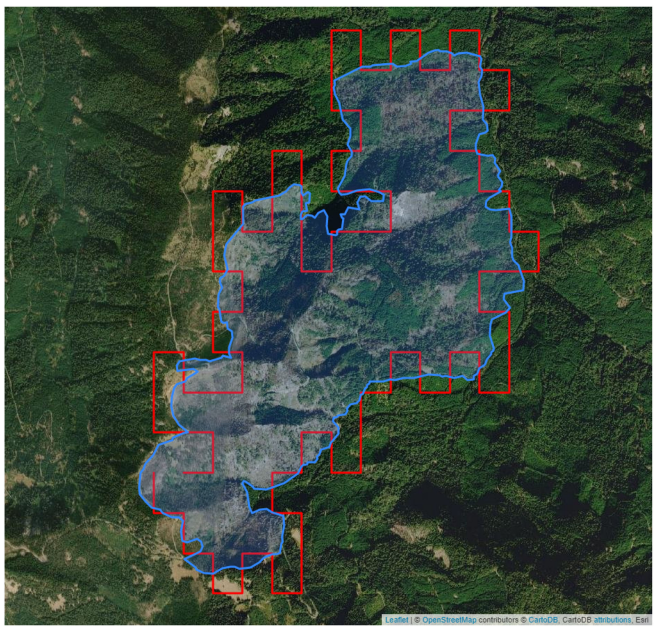

Fig. 6. A simulated CuP sampling path on the 2017 Staley wildfire. The blue region indicates the burned area. The red line indicates the path traveled by the remote sensor when using CuP with a side length of 400 m. Aerial imagery from [39].

mapping the set of sampling locations P_w to a 20 km $\times$ 20 km region centered around each fire. Fig. 6 shows the sampling path used by the CuP algorithm on one such wildfire boundary. We analyze the performance of CuP and S^2 when using regular partitions with side lengths $1/w$ and compare them with the tuned Bang-Bang algorithms that sample at a rate of $1/w$. This corresponds to distances ranging between 200 m and 500 m between samples. We report the average results on 159 different fire boundaries that satisfy Assumption 1. The resulting number of samples, distance traveled, estimation error, and computation time as a function of w for the synthetic and real boundaries are shown in Figs. 4 and 5, respectively.

We make several observations on the results on both real and synthetic boundaries. First, we see that with CuP the number of samples increases linearly with w, as predicted by Theorem 2. We see that the number of samples in S^2 also increases linearly with w, but at a faster rate. The adaptive and fixed Bang-Bang algorithms have similar sample complexity, but both use more samples than both CuP and S^2 for all values of w. This is true for both the synthetic and real boundaries.

Second, the distance traveled by CuP is roughly constant, as predicted by Theorem 3. The distance traveled by the Bang-Bang algorithms is also roughly constant on both the real and synthetic boundaries, but noticeably larger than the distance needed by the equivalent CuP algorithm. In contrast, the distance traveled by S^2 appears to increase linearly with w.

Next, we compute the resulting estimation error to be the proportion of cells in P_w that are mislabeled, which is exactly to the number of cells intersecting the boundary divided by the total number of cells. Both CuP and S^2 find the exact same cut edges, and thus have identical estimation error. The estimation error of the Bang-Bang algorithms is comparable to that of CuP and S^2 on both the real and synthetic data. On the synthetic boundaries, the Bang-Bang algorithms have slightly lower estimation error

TABLE I
TOTAL SAMPLING COSTS ON HISTORICAL WILDFIRE DATA

	Sampling Time (s) Velocity (km/hr)	8 32	8 65	30 32	30 65
Bang-Bang	Total Cost (hr) Error (km^2)	1.83 0.68	1.15 0.68	3.15 0.68	2.47 0.68
S^2	Total Cost (hr) Error (km^2)	6.19 0.68	3.25 0.68	7.28 0.68	2.57 1.13
CuP	Total Cost (hr) Error (km^2)	1.44 0.68	0.90 0.68	2.48 0.68	1.95 0.68

than CuP and S^2, but as discussed above travel farther and require significantly more samples to do so. On the real boundaries, we found that all four algorithms produced results with the exact same estimation error. On these boundaries the most successful Bang-Bang tuning was one with 90^o between samples and no adaptive adjustments. This results in the Bang-Bang algorithms sampling on the same grid as the other algorithms, resulting in identical boundary estimates but requiring more samples than CuP or S^2. In all cases, the figures indicate a clear trade-off of w between linearly increasing sampling cost and the inversely decreasing and estimation error.

Finally, we see that the computation time of CuP and Bang-Bang is several orders of magnitude lower than the computation time of S^2 for our example boundaries. In addition, the computation time of CuP and Bang-Bang appears to scale linearly with w, requiring more time for estimates that contain more cut edges, but providing a constant computation time per sample. In contrast, the computation time of S^2 increases polynomially with respect to w for the same boundaries.

To further illustrate the advantages of CuP over existing methods, we consider the actual costs incurred by sampling the above wildfire boundaries with a particulate matter sensor attached to a UAV. In particular, we consider sampling times of 8 s and 30 s, corresponding to the extremes of the settling time of the Sensirion SPS30 particulate matter sensor [40], and travel times of 32 km/hr and 65 km/hr, corresponding to the velocity range of the DJI Matrice 600 UAV. The total sampling cost is computed as

$$T_{tot} = T_s N + T_t D,$$

where T_s is the time required to obtain a single sample, T_t is the time to travel one unit distance, N is the total number of samples required, and D is the distance traveled.

Table I shows the total sampling cost incurred by the Bang-Bang, S^2, and CuP algorithms. As stated above, the adaptive Bang-Bang steering policy provided no benefit over the non-adaptive Bang-Bang policy on the real fire boundaries, and thus its results are omitted from this table. We report results from each algorithm with the best tuning in terms of average accuracy per unit cost over the 159 fire boundaries. The results show that CuP achieves the same (or lower) estimation error at a lower cost than both the Bang-Bang and S^2 algorithms in all sampling scenarios, providing a 21% reduction in total cost compared to the Bang-Bang policy for all sampling/travel times.

V. CONCLUSION

We have presented a graph-based approach to boundary estimation with mobile sensors. We show that our proposed algorithm uncovers all vertices lying adjacent to the boundary while achieving order-optimal sample complexity and distance traveled while requiring $O(1)$ computation time per sample. To the best of our knowledge, this is the first algorithm that achieves a "zig-zagging" behavior that is both principled and practical for the application of mobile sensing. In our future work, we will consider alternate graph structures, including triangular and hexagonal partitions, as well as non-regular partitions. On the synthetic boundaries, we found that the Bang-Bang algorithm with angle 60^o was most frequently the best tuning in both the adaptive and non-adaptive cases. In this configuration, the Bang-Bang algorithm samples in a hexagonal pattern, indicating that a hexagonal variant of CuP may be of particular interest.

REFERENCES

[1] R. Mur-Artal, J. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.
[2] C. Papachristos, S. Khattak, and K. Alexis, "Uncertainty-aware receding horizon exploration and mapping using aerial robots," in *Proc. - IEEE Int. Conf. Robot. Automat.*, 2017, pp. 4568–4575.
[3] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An efficient sampling-based method for online informative path planning in unknown environments," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1500–1507, Apr. 2020.
[4] Y. Girdhar and G. Dudek, "Optimal online data sampling or how to hire the best secretaries," in *Proc. Can. Conf. Comput. Robot Vis.*, 2009, pp. 292–298.
[5] N. Wilde, D. Kulic, and S. L. Smith, "Bayesian active learning for collaborative task specification using equivalence regions," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1691–1698, Apr. 2019.
[6] W. Luo and K. Sycara, "Adaptive sampling and online learning in multi-robot sensor coverage with mixture of Gaussian processes," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 6359–6364.
[7] D. Jang, J. Yoo, C. Y. Son, D. Kim, and H. J. Kim, "Multi-robot active sensing and environmental model learning with distributed Gaussian process," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5905–5912, Oct. 2020.
[8] J. S. Willners, L. Toohey, and Y. Petillot, "Sampling-based path planning for cooperative autonomous maritime vehicles to reduce uncertainty in range-only localization," *IEEE Robot. Automat. Lett.*, vol. 4, no. 4, pp. 3987–3994, Oct. 2019.
[9] G. Dasarathy, R. Nowak, and X. Zhu, "S2: An efficient graph based active learning algorithm with application to nonparametric classification," in *Proc. Conf. Learn. Theory*, 2015, pp. 503–522.
[10] Y. Wang and A. Singh, "Noise-adaptive margin-based active learning for multi-dimensional data and lower bounds under tsybakov noise," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 2180–2186.
[11] A. Guillory and J. Bilmes, "Average-case active learning with costs," in *Proceeding International Conference on Algorithmic Learning Theory*. Berlin, Heidelberg: Springer, 2009, pp. 141–155.
[12] D. Golovin and A. Krause, "Adaptive submodularity: Theory and applications in active learning and stochastic optimization," *J. Artif. Intell. Res.*, vol. 42, pp. 427–486, 2011.
[13] I. Bogunovic, J. Scarlett, A. Krause, and V. Cevher, "Truncated variance reduction: A unified approach to Bayesian optimization and level-set estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1507–1515.
[14] A. Singh, R. Nowak, and P. Ramanathan, "Active learning for adaptive mobile sensing networks," in *Proc. 5th Inf. Process. Sensor Netw.*, 2006, pp. 60–68.
[15] R. Castro and R. Nowak, "Active learning and sampling," in *Foundations and Applications of Sensor Management*, A.O. Hero, 1st ed. New York, NY, USA: Springer, 2008, Ch. 8.
[16] A. Gotovos, N. Casati, G. Hitz, and A. Krause, "Active learning for level set estimation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2013, pp. 1344–1350.

[17] G. Hitz *et al.*, "Fully autonomous focused exploration for robotic environmental monitoring," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 2658–2664.
[18] J. Lipor, B. P. Wong, D. Scavia, B. Kerkez, and L. Balzano, "Distance-penalized active learning using quantile search," *IEEE Trans. Signal Process.*, vol. 65, no. 20, pp. 5453–5465, Oct. 2017.
[19] D. Wang, J. Lipor, and G. Dasarathy, "Distance-penalized active learning via Markov decision processes," in *Proc. IEEE Data Sci. Workshop*, 2019, pp. 155–159.
[20] J. Hwang, N. Bose, and S. Fan, "AUV adaptive sampling methods: A review," *Appl. Sci.*, vol. 9, no. 15, pp. 1–30, 2019.
[21] D. N. Subramani, P. J. Haley Jr., and P. F. Lermusiaux, "Energy-optimal path planning in the coastal ocean," *J. Geophysical Res.: Oceans*, vol. 122, no. 5, pp. 3981–4003, 2017.
[22] A. Singh, A. Krause, and W. J. Kaiser, "Nonmyopic adaptive informative path planning for multiple robots," in *Proc. Int. Joint Conf. Artif. Intell.*, 2009, pp. 1843–1850.
[23] S. Tschiatschek, A. Singla, and A. Krause, "Selecting sequences of items via submodular maximization," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 2667–2673.
[24] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, "Information-theoretic regret bounds for Gaussian process optimization in the bandit setting," *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 3250–3265, May 2012.
[25] J. Lipor and G. Dasarathy, "Quantile search with time-varying search parameter," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, 2018, pp. 1016–1018, [Online]. Available: http://web.cecs.pdx.edu/%7elipor/Papers/lipor2018quantile.pdf
[26] S. Petillo, H. Schmidt, P. Lermusiaux, D. Yoerger, and A. Balasuriya, "Autonomous & adaptive oceanographic front tracking on board autonomous underwater vehicles," in *Proc. OCEANS -Genova*, 2015, pp. 1–10.
[27] Z. Jin and A. L. Bertozzi, "Environmental boundary tracking and estimation using multiple autonomous vehicles," in *Proc. IEEE Conf. Decis. Control*, 2007, pp. 4918–4923.
[28] A. Joshi, T. Ashley, Y. R. Huang, and A. L. Bertozzi, "Experimental validation of cooperative environmental boundary tracking with on-board sensors," in *Proc. Amer. Control Conf.*, 2009, pp. 2630–2635.
[29] C. Mellucci, P. P. Menon, C. Edwards, and P. G. Challenor, "Environmental feature exploration with a single autonomous vehicle," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 4, pp. 1349–1362, Jul. 2020.
[30] N. J. Gralewicz, T. A. Nelson, and M. A. Wulder, "Spatial and temporal patterns of wildfire ignitions in Canada from1980 to 2006," *Int. J. Wildland Fire*, vol. 21, no. 3, pp. 230–242, 2012.
[31] Y. Zhou, D. R. Obenour, D. Scavia, T. H. Johengen, and A. M. Michalak, "Spatial and temporal trends in lake erie hypoxia, 1987-2007," *Environ. Sci. Technol.*, vol. 47, no. 2, pp. 899–905, 2013.
[32] A. B. Tsybakov *et al.*, "On nonparametric estimation of density level sets," *Ann. Statist.*, vol. 25, no. 3, pp. 948–969, 1997.
[33] C. Scott and R. Nowak, "Minimax-optimal classification with dyadic decision trees," *IEEE Trans. Inf. Theory*, vol. 52, pp. 1335–1353, Apr. 2006.
[34] Y. Xue and Z. J. Zhai, "Inverse identification of multiple outdoor pollutant sources with a mobile sensor," in *Building Simulation*, vol. 10, Tsinghua University Press, Beijing, China: Springer, 2017, pp. 255–263.
[35] K. Shukla, P. Kumar, G. S. Mann, and M. Khare, "Mapping spatial distribution of particulate matter using kriging and inverse distance weighting at supersites of megacity Delhi," *Sustain. Cities Soc.*, vol. 54, 2020, Art. no. 101997.
[36] H.-J. Chu, M. Z. Ali, and Y.-C. He, "Spatial calibration and pm 2.5 mapping of low-cost air quality sensors," *Sci. Rep.*, vol. 10, no. 1, pp. 1–11, 2020.
[37] MTBS *Project: Burned Area Boundaries Dataset*, "USDA Forest Service, U.S. Geological Survey," Apr. 2021. [Online]. Available: https://mtbs.gov/direct-download
[38] C. K. Williams and C. E. Rasmussen, "Covariance Functions" *Gaussian Processes Mach. Learn.*, Cambridge, MA, USA: MIT Press, 2006, ch. 4.
[39] *ESRI world imagery map, Environmental Systems Research Institute*, Dec. 2009. [Online]. Available: https://server.arcgisonline.com/arcgis/rest/services/World_Imagery/MapServer
[40] Sensirion, "Particulate matter sensor SPS30," Sensirion.com. [Online]. Available: https://www.sensirion.com/en/environmental-sensors/particulate-matter-sensors-pm25/ Accessed: Aug. 1, 2021.